FRAME-BASED ALGORITHMS FOR DERIVATIVE-FREE OPTIMISATION

A thesis submitted in fulfilment of the

requirements for the Degree of

Doctor of Philosophy

in Mathematics

by

David Byatt

University of Canterbury

2004

Abstract

A class of provably convergent frame-based line search algorithms that do not explicitly rely on derivative information is developed for unconstrained and linearly constrained optimisation.

A frame-based approach relies on the underlying theory of positive bases. The use of positive bases (either directly or indirectly) facilitates the development of convergence proofs for many (including some already well-established) derivative-free optimisation algorithms. Although positive bases have recently been the subject of renewed interest in optimisation research they do not feature in many modern texts on linear algebra. For this reason background material on positive bases and grids (the precursors to frames) is also presented.

A weakness with a common choice of stopping conditions for existing grid- (which includes pattern search) and frame-based derivative-free methods is identified and a solution is presented. This solution is shown to possess good numerical stability properties when extended for use with existing derivative-based (BFGS and DFP) algorithms—even when approximate second-order information is available to only limited precision—as is usually the case in practice since, for many "real-world" problems, explicit gradient or second order information may not be available and the evaluation of the objective function to arbitrary levels of precision is not possible.

Contents

Notation

a_i	The ith (linear) constraint normal
A	Matrix whose columns are the constraint normals a_i
$\mathcal{A}(x)$	Index set for the active constraints at x
$\mathcal{A}_+(x)$	Superset of $\mathcal{A}(x)$ that includes constraints which are nearly active at x
b_i	Affine translation of constraint c_i
B	Hessian or approximate Hessian matrix
$\mathcal{B}(x, \delta)$	The open ball centred on x with radius δ
c_i	The ith constraint
C	Conjugate factor of an approximate inverse Hessian matrix
$\mathcal{C}^k$	The set of all functions whose kth derivatives exist and are continuous
d	Directional derivatives of the objective function in the conjugate directions
$\bar{d}$	Updated directional derivatives
$\breve{d}$	Distance between grid points on principal axis for diagonally aligned grids
d_i	Distance from the current iterate to constraint c_i
$\mathcal{D}(x)$	Set of feasible directions from x
$\mathcal{D}_L(x)$	Set of linearised feasible directions from x
e_q	The qth Euclidean co-ordinate unit vector
$\mathcal{E}$	Index set for the equality constraints
$\mathfrak{E}$	The set of exterior aligned positive basis directions
f	The objective function $f\colon \mathbb{R}^n \to \mathbb{R}$
$f(x)$	Value of the objective function at the point x
f_j	Value of the objective function at each of the line search steps
$\mathcal{F}$	A frame in $\mathbb{R}^n$
$g^{(k)}$	Abbreviation for $\nabla f(x^{(k)})$
G	Constant Hessian matrix of a quadratic function
$\mathcal{G}$	A grid in $\mathbb{R}^n$
GLM	Abbreviation for grid local minimiser
h	Mesh, grid or frame size parameter
H	Approximation to the inverse Hessian matrix
I	Identity matrix
$\mathcal{I}$	Index set for the inequality constraints
$\mathfrak{I}$	The set of interior aligned positive basis directions
i, j, k	Iteration counters or indices
K, L, M	Positive constants
$\mathcal{K}$	Polyhedral cone of feasible directions
l_*	Location of a line local minimiser
$\mathcal{L}(x, \lambda)$	The Lagrangian

$\mathfrak{L}$	Lipschitz constant
n	Number of independent variables
N	Constant used in defining the sufficient descent parameter $\epsilon = Nh^\nu$
$\mathbb{N}$	The natural numbers
$\mathbb{N}_0$	$\mathbb{N} \cup \{0\}$
m	The cardinality of a positive basis
$o(\cdot)$	Little-oh notation
$\bar{o}(\cdot)$	Bounded little-oh
$\mathcal{O}(\cdot)$	Big-oh notation
$\bar{\mathcal{O}}(\cdot)$	Strictly big-oh
p	A descent or search direction
$\hat{p}$	Feasible direction unit vector
P	Line search trial point immediately on or before l_*
q	A quadratic function
$\mathfrak{q}$	Vector used in the product form of the BFGS update formula
Q	Line search trial point immediately after l_*
QMF	Abbreviation for quasi-minimal frame
QMP	Abbreviation for quasi-minimal point
$\mathbb{R}$	The real numbers
$\mathbb{R}^n$	Euclidean n-space
s	A search direction, usually a member of $\mathcal{S}_+$
$s_{\mathfrak{E}}$	The best search direction from the set of exterior directions
$s_{\mathfrak{I}}$	The best search direction from the set of interior directions
$\mathcal{S}_+$	A positive spanning set; set of candidate search directions
t	Distance from the minimiser to a GLM on the principal axis
$t_{\max}$	Maximum distance from the minimiser to a GLM on the principal axis
u_p	The pth column of U
U	An invertible matrix which maps the constraint normals $a_{i_q} \mapsto e_q$
v	A vector in $\mathbb{R}^n$, usually a member of $\mathcal{V}$
V	Matrix whose columns are members of $\mathcal{V}$
$\mathcal{V}$	Set of vectors which form a basis for $\mathbb{R}^n$
$\mathcal{V}_+$	Set of vectors which form a positive basis for $\mathbb{R}^n$
w	A vector in $\mathbb{R}^n$, usually a member of $\mathcal{W}_+$
$\mathcal{W}_+$	A set of vectors which form a positive basis for a subset of $\mathbb{R}^n$
x	A vector in $\mathbb{R}^n$
$x^{(0)}$	Initial point
$\{x^{(k)}\}$	Sequence of iterates
x^*	Local minimiser
$\breve{x}$	A grid local minimiser, or (quasi-)minimal frame centre
$\bar{x}_{ij}$	Point on the blocking constraint c_i in direction v_j
$\mathcal{X}_m$	Compact set of ordered positive bases with cardinality m
$\{y^{(k)}\}$	Subsequence of iterates

z	Temporary storage vector, usually the point returned by a line search
$\mathfrak{z}$	Difference in the directional derivatives of successive iterations
$\mathbb{Z}$	The set of integers
α	Line search step size parameter
α_j	Line search steps
$\bar{\alpha}_{ij}$	Step size parameter for $\bar{x}_{ij}$
β	Real number contained in the interval $(0, 1)$
γ	Difference in the gradient of successive iterations
δ	Small positive quantity
Δ	Decrease in the objective function due to the search step
$\Delta_{\mathfrak{E}}$	Decrease in the objective function due to exterior direction line search
$\Delta_{\mathfrak{I}}$	Decrease in the objective function due to interior direction line search
ϵ	Sufficient descent parameter
ζ	Real constants
ζ_{ij}	Structure coefficients
η	Integer constants
θ	Various angles
ϑ	Integration dummy variable
Θ	Positive constant
κ	Condition number of Hessian matrix
λ_i	Lagrange multipliers
Λ	Positive constant
$\mu_{\min}, \mu_{\max}$	Minimum and maximum eigenvalues of Hessian matrix
ν	Constant used in defining the sufficient descent parameter $\epsilon = Nh^{\nu}$
ξ, Ξ	Lower and upper bounds on the length of positive basis vectors
ϱ_1, ϱ_2	Lower and upper bounds on successive forward tracking step sizes
$\tau_{\mathcal{A}}$	Tolerance for including constraints in the working set $\mathcal{A}_+$
τ_f	Tolerance for determining the convergence of a sequence of function values
τ_g	Tolerance for determining the convergence of a sequence of gradient values
$\tau_{\text{LS}_1}, \tau_{\text{LS}_2}$	Lower and upper bounds on the line search step lengths
$\tau_{\text{SC}_1}, \tau_{\text{SC}_2}$	Lower and upper bounds on the structure coefficients
τ_V	Lower bound on the minimum value of $\lvert \det(V) \rvert$
τ_x	Tolerance for determining the convergence of a sequence of iterates
τ_θ	Lower bound on the maximum value of direction cosines
$\tau_{\epsilon_1}, \tau_{\epsilon_2}$	Lower and upper bounds for strictly big-oh notation
Φ	Positive constant
Ψ	Positive constant
ω	Various angles
Ω	Feasible region
$\nabla f(x)$	The gradient of f at x
$\nabla^2 f(x)$	The Hessian of f at x
$\emptyset$	The empty set

$\setminus$	Set difference operator: $x \in X \setminus Y \implies x \in X$ and $x \notin Y$.
$\lfloor \cdot \rfloor$	Floor function: $\lfloor x \rfloor$ is the greatest integer less than or equal to x.
$\lceil \cdot \rceil$	Ceiling function: $\lceil x \rceil$ is the least integer greater than or equal to x.
$\| \cdot \|$	Euclidean norm (unless otherwise stated)
$\| \cdot \|_F$	Frobenius norm
$\| \cdot \|_{\mathcal{G}}$	Grid norm (taxi-cab norm)
$\square$	End of proof marker

Chapter 1
Introduction

What is optimisation? The word optimal comes from the Latin word *optimus* which means best [33, p. ix]; so optimisation is the art and science of determining the "best" solutions to certain mathematically defined problems [31, p. 3]. The "goodness" of candidate solutions is determined by an *objective function* whose value depends on a set of independent decision variables or parameters [2, p. 1].

Many optimisation problems arise from modelling physical situations and so there is an extremely wide range of applications [31, p. 3]. In fact optimisation methods reach into almost every area in which numerical information is processed; science, engineering, mathematics, statistics, economics and commerce [31, p. 4]. As Fletcher writes in the preface to [31]:

> The subject of optimization is a fascinating blend of heuristics and rigour, of theory and experiment. It can be studied as a branch of pure mathematics, and yet has applications in almost every branch of science and technology.

An engineer, for example, may be interested in minimising the drag on a new aircraft design whereas an economist may be interested in maximising profits. However since finding the minimiser of a function f is equivalent to finding the maximiser of $-f$, the distinction between minimising and maximising is, mathematically, irrelevant. It is commonplace in the mathematical literature for the words optimise and minimise to be synonymous and this is the meaning used throughout this thesis.

If there are restrictions or constraints that define acceptable values for the decision variables then the optimisation process is called *constrained optimisation*. If there are no such restrictions then the optimisation process is called *unconstrained optimisation*.

In general it is only practicable to find local rather than global solutions to optimisation problems and some conditions such as the continuity of the objective function may need to be imposed in order to guarantee the existence of a minimiser [31, p. 12]. Only local optimisation of continuous functions is considered here.

For the purposes of this thesis, optimisation is the process of finding a local minimiser of an objective function whose independent decision variables may be subject to some constraints. For example, the objective function could represent the cost of production in some commercial or industrial process. The constraints could represent the level of pollution produced, the amount of materials available and the total time until completion. Clearly there are many possible variations on this theme.

Derivative-free optimisation

Consistent use of the terms *derivative-free* and *direct search* when discussing "non-gradient" based optimisation is yet to be properly established. In its broadest sense derivative-free means that the gradient of the objective function is not used. This may be because the gradient is too expensive to calculate, unable to be calculated with sufficient accuracy, or because it simply does not exist.

One of the problems defining derivative-free optimisation can be illustrated with the following example. Quasi-Newton methods are gradient-based methods. They are clearly not derivative-free. However suppose the gradient is approximated by finite differences so that the gradient is not directly evaluated. Now the method relies solely on function values; is it now derivative-free? Some authors say yes, some say no. For completeness definitions of both derivative-free methods and direct search methods are given below. The definition for derivative-free is a more general version of the definition given in [16, p. 399]. However as a precise definition of what is meant by derivative-free is not necessary for the material presented in this thesis the pros and cons of alternative derivative-free definitions will not be discussed further.

Definition 1.1 (Derivative-free). Methods which do not *directly* compute the gradient of the objective function.

Definition 1.2 (Direct search). Methods which use function value comparisons rather than explicit function values [50].

As direct search methods are a subset of derivative-free methods they will not be (explicitly) discussed further (see [45] for a thorough recent survey of direct search methods).

The initial popularity of derivative-free methods began to decline by the 1970s in favour of methods which use derivative information to locate the minimisers of functions. Some authors recommended that methods which did not make use of derivative information be avoided. For example, Gill *et al* [37, p. 93] write:

> A method using function comparison should only be used if there is no other suitable method available. If a user decides to use a function comparison method only because of its simplicity and seeming generality, he may pay a severe price in speed and reliability. ... The substantial disadvantage of function comparison methods is that few (if any) guarantees can be made concerning convergence.

This highlights one of the reasons for the decline in popularity of derivative-free methods: the lack a theoretical convergence and rate of convergence results. Many modern texts on optimisation only mention in passing the existence and use of derivative-free methods. Some authors ignore this type of optimisation method altogether in favour of derivative-based methods. Recently however derivative-free methods have become fashionable again (see for example [16, 29, 50, 61, 77]). This resurgence is due in part to the development of new theoretical frameworks which have allowed the construction of convergence results for some established derivative-free methods (see for example [15, 22, 51, 73, 74]).

Another motivation for the development and use of derivative-free methods is that there is a large number of practical problems where derivative information is just not available. Many optimisation problems do not permit the calculation of analytic derivatives: if the function values are obtained by the measurement of some physical process or some complex numerical simulation for example. The physical nature of some problems may mean that calculating many finite difference approximations is impractical [16, p. 84]; if the objective function is expensive to compute (either in real cost as some expensive physical process, or in the time that such a computation requires) for example. Perhaps calculating an accurate finite difference approximation of the gradient is impractical for physical reasons: it may not be practical to alter the temperature in the chamber where a chemical reaction is taking place by one part in a million and then to measure the effect that this has on the objective function. Optimisation problems of this nature are common in the industrial world. Furthermore if the function values are the result of some measurement then they will also be subject to noise and so derivative information may range from unreliable through to totally unusable. The calibration of robotic arms is one such example from mechanical engineering. Due to the type of measurements recorded during the calibration process there is no derivative information and since the measurements are themselves subject to noise there may even be a loss of continuity of the objective function [18]. Since derivative-free methods do not require derivatives it is often claimed that they are robust for problems where there are discontinuities or where the function values are affected by noise.

Noise. Most optimisation algorithms are not designed to solve problems in which the computation of the function values are subject to noise. It is often assumed that the objective function can be evaluated on a computer to full machine precision. A problem with the optimisation of noisy functions is that if the algorithm requires the estimation

of derivatives by differences and if the difference parameter does not depend on the level of noise then incorrect derivative approximations are usually obtained. This invariably leads to the failure of the algorithm. Another possible problem is that the termination criteria of the algorithm must recognise when differences in the objective function values are only due to noise [54, p. 340]. Since direct search algorithms only rank function values their performance should be relatively unaffected by noise provided the noise level does not alter the ranking of the function values. See [1] for the recent development of a direct search algorithm for the unconstrained optimisation of noisy functions.

Thesis overview

Recent advances (for example [13, 21, 22, 50, 65, 66]) have led to the possibility of efficient, provably convergent, derivative-free optimisation algorithms that are applicable to the solution of large-scale, real-world problems. Previously most provably convergent algorithms made use of gradient information. Recently however derivative-free methods have been developed which are provably convergent and also numerically efficient (for example [23, 67]).

Established techniques for developing and analysing the convergence of optimisation algorithms usually require the exploitation of at least one of three general classical approaches: trust-regions, line searches or grids. Trust-region methods are not considered further here, but see [14] for a very thorough treatment of this topic. A new frame-based approach developed from grid-based methods offers greater flexibility and new opportunities to develop efficient, provably convergent optimisation algorithms.

The main focus of this thesis is the development of a provably convergent frame-based line search algorithm for linearly constrained optimisation. This extends the frame-based line search algorithm for unconstrained optimisation developed in [65]. The extension to handle linear constraints follows the approach presented in [66]. Full convergence results for both simple and sufficient descent algorithms for unconstrained and linearly constrained optimisation are developed. Although convergence has already been established for a simple descent frame-based line search algorithm for unconstrained optimisation the convergence proof presented here is new and less restrictive than the original in [65]. The practical efficiency of these algorithms can be enhanced by avoiding some of the weaknesses of existing grid-based approaches.

Both grids and frames rely on positive bases which are discussed in more detail in the following chapter. Chapter 2 also presents the necessary background on grids and frames. The frame-based line search algorithms for unconstrained and linearly

constrained optimisation are developed in Chapters 3 and 4. The constrained algorithms are automatically applicable to unconstrained optimisation problems however, for clarity, the unconstrained case is developed separately first even though this leads to some duplication.

A weakness with a common choice of stopping conditions for existing grid- and frame-based derivative-free methods is identified and a solution is presented in Chapter 5. Chapter 6 shows that the results developed in Chapter 5 are also effective when extended for use with existing derivative-based (BFGS and DFP) algorithms. This extension is shown to possess good numerical stability properties even when approximate second-order information is available to only limited precision, as is usually the case in practice. The main body of the thesis concludes with a discussion and summary in Chapter 7.

Extensive performance results for specific implementations of algorithms conforming to the frameworks developed are not presented. While this is an important consideration such an investigation at this stage would shift the focus of the research away from the theoretical frameworks and onto specific implementations.

This chapter concludes with a presentation of the background theory for both unconstrained and constrained optimisation. None of this background material is new (see for example [31, 37, 58]). It is included here for completeness and to develop the notation that will be used throughout the remainder of the thesis.

1.1 Unconstrained optimisation

Since unconstrained optimisation is concerned with finding a minimiser of a function it is important to define what is meant by a descent direction. Many authors define this in terms of directional derivatives. However as the material presented in this thesis is to apply in a broader (derivative-free) context a more general definition is used.

Definition 1.3 (Descent direction). A direction $p \in \mathbb{R}^n$ is a descent direction for $f\colon \mathbb{R}^n \to \mathbb{R}$ at a point $x \in \mathbb{R}^n$ if there exists $\delta > 0$ such that $f(x+\alpha p) < f(x)$ whenever $\alpha \in (0, \delta)$.

This definition does not alter established derivative-based notions of descent when derivative information is available. For example, if $f \in \mathcal{C}^1$ (where $\mathcal{C}^k$ denotes the set of all functions whose kth derivatives exist and are continuous) and $p^{\mathsf{T}}\nabla f(x) < 0$ then p is a descent direction for f at x. However if $\nabla f(x) = 0$ then no first-order descent directions for f exist at x. If $f \in \mathcal{C}^2$, $\nabla f(x) = 0$ and $p^{\mathsf{T}}\nabla^2 f(x)\, p < 0$ then p is a (second-order) descent direction for f at x.

For a given objective function $f\colon \mathbb{R}^n \to \mathbb{R}$ the general unconstrained optimisation problem can be written as

$$\begin{aligned} &\text{minimise} && f(x), \\ &\text{subject to} && x \in \mathbb{R}^n. \end{aligned} \tag{1.1}$$

Solutions of (1.1) are local minimisers of f. However, as second-order information may not even exist, stationary points of f are also accepted.

Definition 1.4 (Minimum). If there exists an open ball $\mathcal{B}$ about a point $x^* \in \mathbb{R}^n$ such that

$$f(x^*) \leqslant f(x), \quad \forall x \in \mathcal{B} \tag{1.2}$$

then $f(x^*)$ is a (local) minimum of f.

Note that in order to refer to the open ball centred on x_0 with radius δ the notation $\mathcal{B}(x_0, \delta) = \{x \in \mathbb{R}^n : \|x - x_0\| < \delta\}$ will be used. Unless otherwise stated $\|\cdot\|$ represents the Euclidean norm.

Definition 1.5 (Minimiser). A point $x^* \in \mathbb{R}^n$ for which inequality (1.2) holds is called a (local) minimiser of f.

If $f(x^*) \leqslant f(x)$ for all $x \in \mathbb{R}^n$ then x^* is a *global* minimiser and $f(x^*)$ is the global minimum of f. If there exists an open ball $\mathcal{B}$ about x^* such that inequality (1.2) strictly holds for all $x \in \mathcal{B} \backslash \{x^*\}$ then x^* is a *strict* local minimiser of f and $f(x^*)$ is the strict minimum. If there exists an open ball $\mathcal{B}$ about x^* such that inequality (1.2) holds and x^* is the only minimiser in $\mathcal{B}$ then x^* is an *isolated* minimiser.

If $f \in \mathcal{C}^1$ then by Taylor series

$$f(x + p) = f(x) + p^{\mathsf{T}} \nabla f(x) + o(\|p\|). \tag{1.3}$$

If $\nabla f(x)$ is non-zero then $p = -\nabla f(x)$ is a first-order descent direction for f at x and so x is not a local minimiser of f. Therefore

$$\nabla f(x^*) = 0 \tag{1.4}$$

is a necessary first-order condition for x^* to be a solution of the unconstrained optimisation problem (1.1).

Similarly, if $f \in \mathcal{C}^2$ then

$$f(x + p) = f(x) + p^{\mathsf{T}} \nabla f(x) + \tfrac{1}{2} p^{\mathsf{T}} \nabla^2 f(x)\, p + o(\|p\|^2). \tag{1.5}$$

If $\nabla f(x) = 0$ and $p^{\mathsf{T}}\nabla^2 f(x)\,p < 0$ then p is a descent direction for f at x so that x is not a local minimiser. Hence a necessary second-order condition for x^* to be a local minimiser is that $p^{\mathsf{T}}\nabla^2 f(x^*)\,p \geqslant 0$ for every $p \in \mathbb{R}^n$ so that $\nabla^2 f(x^*)$ is positive-semi-definite. If equation (1.4) holds and $p^{\mathsf{T}}\nabla^2 f(x^*)\,p > 0$ for all non-zero p then by equation (1.5) there exists a $\delta > 0$ such that $f(x^*) \leqslant f(x)$ for all $x \in \mathcal{B}(x^*, \delta)$. Hence a sufficient second-order condition for x^* to be a local minimiser is that $\nabla^2 f(x^*)$ is positive-definite.

1.2 Constrained optimisation

In many practical situations there are restrictions on the acceptable solutions for the optimisation problem (1.1): one or more of the variables may represent physical quantities (perhaps the amount of material required for some manufacturing process for example) and so must remain non-negative. Typically these restrictions or *constraints* are characterised by a set of functions $\{c_i(x)\}$ and split into two categories: equality constraints and inequality constraints. The general constrained optimisation problem can then be written as

$$\begin{aligned} \text{minimise} \quad & f(x), \\ \text{subject to} \quad & c_i(x) = 0, \quad \forall i \in \mathcal{E} \\ & c_i(x) \geqslant 0, \quad \forall i \in \mathcal{I} \end{aligned} \tag{1.6}$$

where $\mathcal{E}$ and $\mathcal{I}$ are the (finite and disjoint) index sets for the equality and inequality constraints.

Definition 1.6 (Equality constraints). The equality constraints are the constraints $c_i(x)$ for which an acceptable point $x \in \mathbb{R}^n$ must satisfy $c_i(x) = 0$ for each $i \in \mathcal{E}$.

Definition 1.7 (Inequality constraints). The inequality constraints are the constraints $c_i(x)$ for which an acceptable point $x \in \mathbb{R}^n$ must satisfy $c_i(x) \geqslant 0$ for each $i \in \mathcal{I}$.

Definition 1.8 (Feasible point). A point $x \in \mathbb{R}^n$ is feasible if it satisfies all of the constraints. That is $c_i(x) = 0$ for each $i \in \mathcal{E}$ and $c_i(x) \geqslant 0$ for each $i \in \mathcal{I}$.

Definition 1.9 (Feasible region). The feasible region Ω is the set of all feasible points.

The minima and minimisers for constrained optimisation problems are defined in a similar way to those in Definitions 1.4 and 1.5 for the unconstrained case with the addition that points must be feasible. Hence the open ball $\mathcal{B}$ of inequality (1.2) is replaced by $\mathcal{B} \cap \Omega$ for the constrained optimisation problem. Similarly, if $f(x^*) \leqslant f(x)$ for all $x \in \Omega$, then $x^* \in \Omega$ is a global constrained minimiser with global minimum $f(x^*)$.

Five more definitions are presented before developing the necessary conditions for x^* to be a solution of the constrained optimisation problem (1.6) when the constraints are linear.

Definition 1.10 (Feasible step). If x and $x + p$ are both feasible points then p is a feasible step from x.

Definition 1.11 (Active constraint). If $c_i(x) = 0$ then c_i is active at x. The index set for all active constraints at a point x is given by $\mathcal{A}(x) = \{i \in \mathcal{E} \cup \mathcal{I} : c_i(x) = 0\}$.

Definition 1.12 (Feasible direction). A unit vector $\hat{p} \in \mathbb{R}^n$ is a feasible direction from a feasible point x if there exists an infinite sequence of feasible points $\{x^{(k)}\} \subseteq \Omega$ such that $x^{(k)} \to x$ as $k \to \infty$ and

$$\frac{x^{(k)} - x}{\|x^{(k)} - x\|} \to \hat{p} \text{ as } k \to \infty.$$

The set of all feasible directions from x is denoted by $\mathcal{D}(x)$. When non-linear constraints are present it is often more convenient to consider the set of linearised feasible directions.

Definition 1.13 (Linearised feasible directions). The set $\mathcal{D}_L(x)$ of linearised feasible directions from a feasible point x is the set of all unit vectors $\hat{p} \in \mathbb{R}^n$ such that $\hat{p}^{\mathsf{T}} \nabla c_i(x) = 0$ for all $i \in \mathcal{E}$ and $\hat{p}^{\mathsf{T}} \nabla c_i(x) \geqslant 0$ for all $i \in \mathcal{I} \cap \mathcal{A}(x)$.

It is shown in [31, p. 202] that $\mathcal{D}(x) \subseteq \mathcal{D}_L(x)$. Points at which $\mathcal{D}(x) = \mathcal{D}_L(x)$ are called *regular* points. The definitions presented so far are applicable for general constraints. However only linear constraints are considered throughout the remainder of this thesis. Note that if the constraints are linear then $\mathcal{D}(x) = \mathcal{D}_L(x)$ for all $x \in \Omega$ and so all feasible points are regular points [31, p. 203].

Linear constraints. If the constraints in the general constrained optimisation problem (1.6) are all linear then the resulting problem is a linearly constrained optimisation problem. A point x^* is a solution of this problem if it satisfies the Karush-Kuhn-Tucker (or KKT) conditions:

$$x^* \in \Omega, \tag{1.7}$$

$$\nabla f(x^*) = \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i \nabla c_i(x^*), \tag{1.8}$$

$$\lambda_i c_i(x^*) = 0, \quad \forall i \in \mathcal{E} \cup \mathcal{I}, \tag{1.9}$$

$$\lambda_i \geqslant 0, \quad \forall i \in \mathcal{I}. \tag{1.10}$$

Equation (1.9) is sometimes referred to as the *complimentarity condition*. This forces $\nabla f(x^*) = 0$ to be a necessary first-order optimality condition whenever the constraints are inactive at x^*. If exactly one of λ_i or $c_i(x^*)$ but not both is zero then *strict* complimentarity is said to hold at x^*. If λ_i and $c_i(x^*)$ are both zero the constraint c_i is said to be *weakly active* at x^* [31, p. 201]. Note that at regular points the KKT conditions (1.7)–(1.10) are also applicable for generally (that is non-linearly) constrained optimisation problems [31, p. 200]. Sometimes the KKT conditions are stated in terms of the *Lagrangian* $\mathcal{L}(x, \lambda)$ defined as

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(x)$$

so that a solution x^* of the linearly constrained optimisation problem must satisfy $\nabla_x \mathcal{L}(x^*, \lambda) = 0$. The constants λ_i are known as the *Lagrange multipliers*. If the active constraints at x are linearly independent then the Lagrange multipliers at x are unique [31, p. 196].

A necessary second-order condition for x^* to be a solution of the general linearly constrained optimisation problem is that $\nabla^2_{xx} \mathcal{L}(x^*, \lambda)$ is positive-semi-definite on the subspace orthogonal to the active constraint normals (i.e. on the intersection of the active constraint tangent hyperplanes). If strict complimentarity holds then a sufficient condition is that $\nabla^2_{xx} \mathcal{L}(x^*, \lambda^*)$ is positive-definite on that subspace [31, pp. 210–211].

Note that although constraints which are simple bounds are a subset of linear constraints there exist special techniques for solving such problems more efficiently (for more details see for example [31, 37, 58]).

Linear programming

If the objective function is linear and the constraints are linear then the resulting optimisation problem is known as a linear programming problem or linear program. Management science and operations research make extensive use of linear models and so linear programming is particularly important in these areas [58, p. 6]. Special techniques have been developed that make use of the special structure of linear programming problems in order to improve the efficiency with which they are solved. This particular specialisation is not part of the scope of the research presented here and so linear programming will not be considered further in this thesis.

Quadratic programming

If the objective function is quadratic and the constraints are linear then the resulting optimisation problem is known as a quadratic programming problem or quadratic program. Quadratic programs can either be solved, or shown to be infeasible, in a finite number of steps. The number of such steps however depends on the nature of the objective function and the number of inequality constraints. If the Hessian of the objective function is positive-semi-definite then a solution x^* is a global solution (by the convexity of the objective function and feasible region) and the resulting quadratic program is called a *convex* quadratic program; which in some cases is no more difficult to solve than a linear program. If the Hessian is positive-definite then the global solution x^* is also unique. Quadratic programs in which the Hessian of the objective function is indefinite tend to be more difficult to solve as they can have several stationary points [58, pp. 441–442]. In some cases a quadratic program may have a special structure which can be utilised in order to gain improved efficiency or stability [31, pp. 247–250]. However, like linear programming, quadratic programming will not be considered further here.

Chapter 2

Positive bases, grids, and frames

Positive bases are a key feature of both grid- and frame-based optimisation methods. As shown in [22], grid-based methods are a generalisation of the pattern search methods discussed in [29, 73, 74] and frames are an extension of grids [21, 65, 66]. The flexibility of grid- and especially frame-based methods means that many existing derivative-free algorithms can be formulated in terms of grids or frames. Such a formulation provides a theoretical framework which guarantees the convergence of many derivative-free optimisation algorithms. This is important since many of the early derivative-free methods developed in the 1950s and 1960s were considered by some to be ad hoc methods. Without derivative information their convergence was considered unprovable (see the quote on page 2). The direct search method of Hooke and Jeeves [44] for example was developed in 1961 and yet its convergence properties were not widely known for over 30 years. However, as the method conforms to a grid-based framework, its convergence is assured [22, 23]. Furthermore, algorithms which do not appear to conform to grid or frame-based approaches are sometimes able to be modified to fit such a framework thus guaranteeing their convergence. One such example is the simplex method of Nelder and Mead [57] developed in 1965. Despite its age the Nelder-Mead simplex method is still the method of choice for many practitioners in the fields of statistics, engineering and the physical and medical sciences. Although it is popular and enjoys frequent practical success the convergence results developed so far only apply in one dimension [46]. A family of strictly convex $\mathcal{C}^2$ functions are known for which the Nelder-Mead algorithm converges to non-stationary points when started with a particular initial configuration [53]. At present there is no function in any dimension greater than one for which the Nelder-Mead algorithm has been proved to converge to a minimiser [46, pp. 113–114]. However the algorithm is easily adaptable to fit within a frame-based framework which improves its performance and guarantees convergence whilst maintaining the "spirit" of the original algorithm [8, 67].

The algorithms for unconstrained and linearly constrained optimisation developed in Chapters 3 and 4 are all frame-based algorithms. However since frames have been developed from grids it is important to investigate the behaviour of grid-based algorithms to avoid some of their weaknesses. One area in particular where derivative-free methods have a potential weakness is deciding suitable stopping conditions. This is investigated further in Chapter 5.

2.1 Positive bases

The theory of positive bases was introduced by Davis [26] in 1954. Although it does not appear in many modern texts on linear algebra the theory of positive bases has recently been the subject of renewed interest within the optimisation community (see for example [23, 29, 47, 48, 78]).

Definition 2.1 (Positive basis). A positive basis $\mathcal{V}_+$ for $\mathbb{R}^n$ is a set of vectors for which the following conditions hold:

(a) Every vector in $\mathbb{R}^n$ can be written as a non-negative combination of the vectors in the positive basis.

(b) No element of $\mathcal{V}_+$ is expressible as a non-negative combination of the remaining elements of $\mathcal{V}_+$.

Finite sets which satisfy (a), but not necessarily (b), are called positive spanning sets. Property (a) is the crucial one but positive bases are usually preferred for reasons of economy [63]. It is shown in [26] that, unlike bases for $\mathbb{R}^n$ which have exactly n elements, the cardinality $|\mathcal{V}_+|$ of a positive basis for $\mathbb{R}^n$ satisfies

$$n+1 \leqslant |\mathcal{V}_+| \leqslant 2n.$$

Example 2.1 (Minimal positive basis). If $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a basis for $\mathbb{R}^n$ then

$$\mathcal{V}_+ = \{v_1, \ldots, v_n, -\sum_{i=1}^{n} v_i\}$$

is a positive basis for $\mathbb{R}^n$ with $n+1$ elements.

Example 2.2 (Maximal positive basis). If $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a basis for $\mathbb{R}^n$ then

$$\mathcal{V}_+ = \{\pm v_1, \ldots, \pm v_n\}$$

is a positive basis for $\mathbb{R}^n$ with $2n$ elements.

Maximal positive bases are useful for forming second-order finite difference estimates of gradients. Minimal positive bases can be used to provide a first-order estimate of the gradient together with an estimate of its accuracy [23]. Other examples and properties of positive bases can be found in [26].

Most existing convergence proofs for derivative-free methods that use positive bases rely either explicitly ([21, 22]) or implicitly ([29]) on ordered positive bases and the

structural equivalence of a subsequence of positive bases. These constructs overcome some of the problems that arise when proving convergence: in particular, guaranteeing the existence of the limit of a sequence of positive bases. Certain extra assumptions are required if this limit is also to be a positive basis. The convergence proofs for the frame-based algorithms presented in Chapters 3 and 4 are more general and do not rely on ordered positive bases or structural equivalence. However since many existing results do rely on these ideas they are included for completeness.

Definition 2.2 (Ordered positive bases). If the basis $\mathcal{V}$ for $\mathbb{R}^n$ is extended to create the positive basis $\mathcal{V}_+$ it is often assumed that the first n members of $\mathcal{V}_+$ are those of $\mathcal{V}$ (in the same order) and that the remaining $|\mathcal{V}_+| - n$ members of $\mathcal{V}_+$ are a linear combination of the members of $\mathcal{V}$ so that

$$v_j = \sum_{i=1}^{n} \zeta_{ij} v_i \tag{2.1}$$

for each $j \in \{n+1, \ldots, |\mathcal{V}_+|\}$, where each $\zeta_{ij} \in \mathbb{R}$. Equation (2.1) requires the members of $\mathcal{V}_+$ to assume a specific order. Positive bases satisfying these requirements are called ordered positive bases [21, 22].

Definition 2.3 (Structural equivalence). Two ordered positive bases $\{v_i\}_{i=1}^m$ and $\{w_i\}_{i=1}^m$ are structurally equivalent if

$$v_j = \sum_{i=1}^{n} \zeta_{ij} v_i \iff w_j = \sum_{i=1}^{n} \zeta_{ij} w_i, \quad \forall j \in \{n+1, \ldots, m\}.$$

Structurally equivalent positive bases necessarily have the same cardinality [22]. The quantities ζ_{ij} are sometimes referred to as the *structure coefficients* [21]. Structural equivalence is useful in that if the set $\{v_i\}_{i=1}^n$ is a basis for $\mathbb{R}^n$ and these vectors are also conjugate directions of an estimated Hessian matrix then the associated positive basis automatically takes into account second order information [64]. The usefulness of conjugate directions for grid- and frame-based optimisation methods will be discussed in more detail in Chapter 5.

Definition 2.4 (Limit of a sequence of ordered positive bases). A set $\{v_i^{(\infty)}\}_{i=1}^m$ with cardinality m is the limit of a sequence of ordered positive bases $\mathcal{V}_+^{(k)} = \{v_i^{(k)}\}_{i=1}^m$ with cardinality m if

$$\lim_{k \to \infty} v_i^{(k)} = v_i^{(\infty)}, \quad \forall i \in \{1, \ldots, m\}.$$

Extra assumptions are required to ensure that the limit (if it exists) is also a positive basis. In particular it is necessary to prevent the span of the limiting set of vectors from collapsing to a subspace of $\mathbb{R}^n$. This is discussed in more detail in [21, 22, 65, 67] and the following section. Limits of sequences of ordered positive bases which are themselves ordered positive bases are used to prove the convergence of the general class of grid- and frame-based optimisation algorithms in [21, 22].

Non-degenerate sequences of positive bases

Positive bases are useful in an optimisation context because if the gradient of the objective function is non-zero then at least one of the positive basis directions is a descent direction. A non-degenerate sequence of positive bases is one in which there is a positive basis direction which is not (and in the limit does not become) an arbitrarily poor descent direction. In practice however the gradient may not be known so that appropriately orientating the positive basis is difficult: unless there is a positive uniform lower bound on the maximum direction cosine between *any* non-zero vector in $\mathbb{R}^n$ and the positive basis directions. To enforce this condition most grid- and frame-based algorithms have used (or assumed) ordered positive bases. For such positive bases if the set of vectors $\mathcal{V} = \{v_i\}_{i=1}^n$ is a basis for $\mathbb{R}^n$ and $V = [v_1, \ldots, v_n]$ is the matrix whose columns are the elements of $\mathcal{V}$, then typically, the following conditions are required to ensure that the limit of a sequence of such ordered positive bases is also a positive basis:

$$\text{(a)} \qquad |\det(V)| \geqslant \tau_V \tag{2.2}$$

$$\text{(b)} \qquad \|v\| \leqslant \Xi, \quad \forall v \in \mathcal{V} \tag{2.3}$$

for some (small) positive constant τ_V and some (large) positive constant Ξ. These two conditions on their own however do not guarantee a non-degenerate sequence of positive bases as the following example illustrates.

Example 2.3. Let $\mathcal{V} = \{v_1, v_2\}$ be a basis for $\mathbb{R}^2$ and $V = [v_1, v_2]$ be the associated matrix whose columns are the positive basis directions. Then $|\det(V)|$ is the area of the parallelogram spanned by v_1 and v_2 so that $|\det(V)| = \|v_1\| \cdot \|v_2\| \cdot |\sin\theta|$ where θ is the angle between v_1 and v_2. Hence equations (2.2) and (2.3) require that each $\|v_i\|$ is bounded away from zero and that $\tau_\theta \leqslant \theta \leqslant \pi - \tau_\theta$, for some typically small positive constant τ_θ. Figure 2.1 shows basis directions v_1 and v_2 which satisfy conditions (2.2) and (2.3). However to create a positive basis $\mathcal{V}_+$ with three members it is only necessary to include the vector v_3 inside the shaded region of Figure 2.1. If $v_3 = -\zeta v_1 - v_2$ for

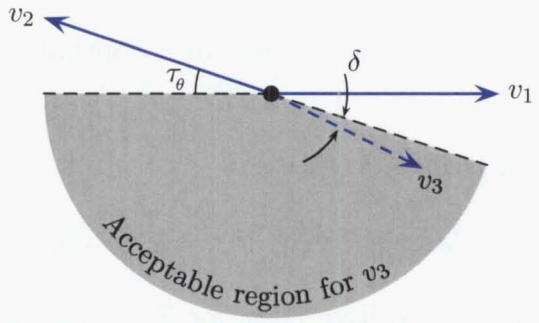

Figure 2.1: Wide angle basis.

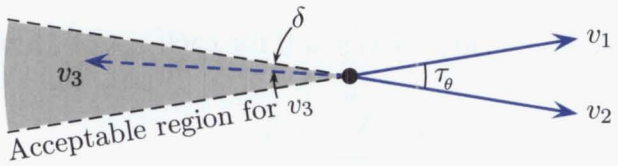

Figure 2.2: Narrow angle basis.

some $\zeta > 0$ then v_3 completes the positive basis. However if a sequence of such bases is formed with $v_3^{(k)} = -\zeta^{(k)} v_1^{(k)} - v_2^{(k)}$ then if $\zeta^{(k)} \to 0$ the descent along each of the positive basis directions may become arbitrarily poor; the angle δ (shown in Figure 2.1) tends to zero as $\zeta^{(k)} \to 0$. A similar situation is illustrated in Figure 2.2 where the angle between vectors v_1 and v_2 attains the lower bound τ_θ.

Requiring a sequence of such positive bases to be structurally equivalent avoids the potential problem of arbitrarily poor descent directions. The structural equivalence of a sequence of positive bases requires that the structure coefficients are all independent of k. For such a sequence the values of $\zeta^{(k)}$ are fixed and so cannot tend to zero. Placing bounds on acceptable values of the structure coefficients also prevents arbitrarily poor descent directions.

In higher dimensions $|\det(V)|$ is the (hyper-)volume of the parallelepiped defined by the basis vectors v_i. The lower bound on $|\det(V)|$ prevents this volume from collapsing and this prevents the basis vectors from lying in a subspace of $\mathbb{R}^n$. It also places the implicit lower bound of $\tau_V \Xi^{1-n}$ on the lengths of the basis vectors [65]. However if one of the basis vectors attains this lower bound then all of the remaining basis vectors must have the maximum length Ξ. If it is necessary to have a lower bound that all of the basis vectors can achieve then the lower bound increases (for typical values of τ_V and Ξ) to $\tau_V^{(1/n)}$ which tends to one as n tends to infinity for any fixed positive

constant τ_V. The usefulness of positive bases in an optimisation context stems from the following result [23].

Theorem 2.1. *If $\mathcal{V}_+$ is a positive basis then*

$$v^T g \geqslant 0, \quad \forall v \in \mathcal{V}_+ \implies g = 0.$$

Proof. Let $\mathcal{V}_+ = \{v_i\}_{i=1}^m$ be a positive basis for $\mathbb{R}^n$. Since any vector in $\mathbb{R}^n$ can be written as a non-negative combination of the positive basis vectors let

$$-g = \sum_{i=1}^{m} \zeta_i v_i$$

where $\zeta_i \geqslant 0$ for each $i \in \{1, \ldots, m\}$. If $v_i^T g \geqslant 0$ for each $v_i \in \mathcal{V}_+$ then

$$0 \leqslant \sum_{i=1}^{m} \zeta_i v_i^T g$$

and since $-g^T g \leqslant 0$ it follows that

$$0 \leqslant \sum_{i=1}^{m} \zeta_i {v_i}^T g = -g^T g \leqslant 0$$

and so the only possibility is $g = 0$. □

The importance of this result will be expanded upon in the following section on grids.

2.2 Grids

The grid $\mathcal{G}(x_0, h, \mathcal{V})$ is the (infinite) set of points in $\mathbb{R}^n$ defined by a point x_0 on the grid, a grid size parameter $h > 0$ and a set of n linearly independent basis vectors $\mathcal{V} = \{v_i\}_{i=1}^n$ so that

$$\mathcal{G}(x_0, h, \mathcal{V}) = \left\{ x \in \mathbb{R}^n \, : \, x = x_0 + h \sum_{i=1}^{n} \eta_i v_i, \; \forall \eta_i \in \mathbb{Z} \right\}.$$

When the values of the parameters x_0, h and $\mathcal{V}$ are not in question the grid may be abbreviated as $\mathcal{G}$. The grid size parameter h is adjusted from time to time in order to ensure that successive grids become finer in a manner needed to establish convergence [22]. Convergence is shown with the use of Theorem 2.1 which leads to the following definition [22].

Definition 2.5 (Grid local minimiser). A point $\breve{x} \in \mathcal{G}$ is a grid local minimiser for the function f on the grid $\mathcal{G}$ with respect to a positive basis $\mathcal{V}_+$ if

$$f(\breve{x} + hv) \geqslant f(\breve{x}), \quad \forall v \in \mathcal{V}_+.$$

The term grid local minimiser will be abbreviated as GLM. For grid-based methods the points $\breve{x} + hv$ must also lie on the grid for each $v \in \mathcal{V}_+$; this can be guaranteed if each member of $\mathcal{V}_+$ is an integer linear combination of the members of $\mathcal{V}$ [23]. The definition of a GLM is motivated by the fact that if $\mathcal{V}_+$ is a positive basis then by Theorem 2.1,

$$v^{\mathsf{T}} \nabla f(x) \geqslant 0, \quad \forall v \in \mathcal{V}_+ \implies \nabla f(x) = 0.$$

The conditions which define a GLM are a finite difference approximation to this [22]. At a GLM there is sufficient information to give an approximation to $\nabla f(x)$ or at least an upper bound on $\|\nabla f(x)\|$ [29] (see [23, 37, 58] for details on the practical problem of estimating derivative information using function values at grid points). The flexibility of grid-based methods is summarised in [22] as follows:

> An important aspect of the main convergence result is that successive grids may be arbitrarily translated, rotated, and sheared relative to one another, and each grid axis may be rescaled independently of the others. This flexibility allows second-order information to be incorporated into the shape of successive grids, for example, by aligning grid axes along conjugate directions or the principal axes of an approximating quadratic. The hope is to construct nonderivative algorithms that possess useful properties of conjugate direction or quasi-Newton algorithms, thus exploiting curvature information without assuming the existence of second derivatives or the availability of first derivatives.

Grid-based methods are not as restrictive as the pattern search methods of [47, 74] where a single set of grid axes is used, only rational scalings of the grids are allowed and arbitrary translations are not permitted. Furthermore the pattern search methods of [47, 74] do not allows scalings and re-alignments to reflect curvature information [22]. More details on the differences between grid-based and pattern search methods can be found in [22]. A simple example of a grid-based algorithm is now presented.

Grid-based algorithms

Template G1, below, is a simplified version of a grid-based algorithm that appears in [22]. The algorithm generates a sequence of GLMs on a succession of progressively

finer grids. This is achieved by evaluating the objective function f at each of the nearby grid points $x^{(k)} + h^{(k)}v$, for each $v \in \mathcal{V}_+^{(k)}$. If no grid points with function values lower than $f(x^{(k)})$ are found then the current iterate is a GLM. If points with function values lower than $f(x^{(k)})$ exist then the next iterate is set to the point with the lowest function value. Each time a GLM is found the grid size parameter $h^{(k)}$ is reduced and a new grid is created about the iterate with the lowest function value. This process is repeated (typically until the grid size parameter has been reduced below a certain cut-off tolerance).

Template G1: Generic grid-based algorithm

Step 1: Set $j = 0$, $k = 0$ and choose the initial point $x^{(0)}$.

Step 2: Choose $h^{(k)}$, $\mathcal{V}^{(k)}$ and $\mathcal{V}_+^{(k)}$ to determine the grid $\mathcal{G}^{(k)} = \mathcal{G}(x^{(k)}, h^{(k)}, \mathcal{V}^{(k)})$.

Step 3: Evaluate $f(x^{(k)})$ and $f(x^{(k)} + h^{(k)}v)$ for each $v \in \mathcal{V}_+^{(k)}$.

Step 4: Set $x^{(k+1)}$ to be the lowest point found in Step 3.

Step 5: If $x^{(k)}$ is a GLM then:

Set $\breve{x}^{(j)} = x^{(k)}$, increment j and k.

If the stopping conditions are satisfied then stop, otherwise go to Step 2.

Otherwise:

Increment k and go to Step 3.

With the appropriate conditions it can be shown (provided the stopping conditions are never invoked and $h^{(k)}$ tends to zero as k tends to infinity) that the sequence $\{\breve{x}^{(j)}\}$ of GLMs is infinite and the cluster points of $\{\breve{x}^{(j)}\}$ are stationary points of f. Furthermore as the sequence of function values $\{f(x^{(k)})\}$ decreases monotonically all cluster points have the same function value. The convergence of grid-based methods is established in [22].

2.3 Frames

Frames are more flexible, cut-down versions of grids. A frame is a set of points in $\mathbb{R}^n$ defined by a frame centre, a frame size parameter and a positive basis, without the need for an underlying grid. The frame $\mathcal{F}(x_0, h, \mathcal{V}_+)$ is the set of at most $2n$ points defined by the central frame point x_0, frame size parameter $h > 0$ and positive basis $\mathcal{V}_+$ so that

$$\mathcal{F}(x_0, h, \mathcal{V}_+) = \{x_0 + hv : v \in \mathcal{V}_+\}.$$

As with grids, when the values of the parameters x_0, h, and $\mathcal{V}_+$ are not in question, the frame may be abbreviated as $\mathcal{F}$. For a given positive basis and frame size it is possible to refer to the frame centred on x_0, or the frame about x_0. The frame size parameter h is adjusted from time to time in a manner that guarantees convergence under the appropriate conditions. Frame-based methods can generate iterates which are equivalent to the GLMs of grid-based methods. Such iterates are called minimal frame centres [21].

Definition 2.6 (Minimal frame centre). A point $\breve{x} \in \mathbb{R}^n$ is a minimal frame centre for the function f with respect to a positive basis $\mathcal{V}_+$ if

$$f(\breve{x} + hv) \geqslant f(\breve{x}), \quad \forall v \in \mathcal{V}_+.$$

If $\breve{x}$ is a minimal frame centre then the associated frame is called a minimal frame about $\breve{x}$ or more simply, a minimal frame [21]. For algorithms that enforce some measure of sufficient descent convergence can be established for a sequence of frame centres which are "nearly" minimal, or more formally, ϵ-*quasi-minimal* where ϵ is some sufficient descent parameter [21].

Definition 2.7 (ϵ-quasi-minimal frame centre). A point $\breve{x} \in \mathbb{R}^n$ which satisfies the weaker condition

$$f(\breve{x} + hv) \geqslant f(\breve{x}) - \epsilon, \quad \forall v \in \mathcal{V}_+$$

for some $\epsilon > 0$ is an ϵ-quasi-minimal frame centre.

When the value of ϵ is not in question the term quasi-minimal frame centre may be used. The terms ϵ-quasi minimal frame and quasi-minimal frame will be abbreviated as ϵ-QMF and QMF. The terms ϵ-quasi minimal frame centre and quasi-minimal frame centre will be abbreviated as ϵ-QMP and QMP (quasi-minimal point). The sufficient descent parameter ϵ determines whether the objective function can be reduced by a sufficient amount over the frame. If a frame is not ϵ-quasi-minimal then the value of the objective function can be reduced by more than ϵ by moving from the frame centre to one of the frame points. If such a reduction is not possible then the central frame point is an ϵ-QMP and the frame is an ϵ-QMF.

Frame-based algorithms

Template F1, below, is a simplified version of a frame-based algorithm that appears in [21]. The algorithm generates a sequence of QMPs. This is achieved by evaluating the objective function at each of the adjacent positive basis points $x^{(k)} + h^{(k)}v$ for

each $v \in \mathcal{V}_+^{(k)}$. The next iterate is set to the point with the lowest function value. If no points with function values lower than $f(x^{(k)}) - \epsilon^{(k)}$ are found then the current iterate is an $\epsilon^{(k)}$-QMP. Each time an $\epsilon^{(k)}$-QMF is found the minimum frame size parameter $h_{\min}^{(k)}$ is reduced and a new frame is created about the iterate with the lowest function value. This process is repeated (typically until the frame size parameter has been reduced below a certain cut-off tolerance).

To establish convergence, sufficient descent frame-based methods require that the sufficient descent parameter $\epsilon^{(k)} = o(h^{(k)})$ so that if $h^{(k)}$ tends to zero as k tends to infinity then $\epsilon^{(k)}/h^{(k)}$ also tends to zero as k tends to infinity. In practice a useful choice for $\epsilon^{(k)}$ is

$$\epsilon^{(k)} = N(h^{(k)})^{\nu}$$

for some positive constant N (that may be adjusted from time to time) and $\nu > 1$ [21]. This choice of ϵ was used in the frame-based convergent variant of the Nelder-Mead algorithm in [67] and is also used in the frame-based algorithm presented in Template F1.

Template F1: Generic frame-based algorithm

Step 1: Set $j = 0$ and $k = 0$. Choose $N_{\max} > 0$, $\nu > 1$ and the initial point $x^{(0)}$.

Step 2: Choose $0 < N^{(k)} \leqslant N_{\max}$ and $h_{\min}^{(k)} > 0$.

Step 3: Choose $\mathcal{V}_+^{(k)}$ and $h^{(k)} \geqslant h_{\min}^{(k)}$ to determine the frame $\mathcal{F}^{(k)} = \mathcal{F}(x^{(k)}, h^{(k)}, \mathcal{V}_+^{(k)})$.

Step 4: Calculate $\epsilon^{(k)} = N^{(k)}(h^{(k)})^{\nu}$.

Step 5: Evaluate $f(x^{(k)})$ and $f(x^{(k)} + h^{(k)}v)$ for each $v \in \mathcal{V}_+^{(k)}$.

Step 6: Set $x^{(k+1)}$ to the lowest point found in Step 5.

Step 7: If $\mathcal{F}^{(k)}$ is $\epsilon^{(k)}$-QMF then:

Set $\breve{x}^{(j)} = x^{(k)}$, increment j and k.

If the stopping conditions are satisfied then stop, otherwise go to Step 2.

Otherwise:

Increment k and go to Step 3.

With the appropriate conditions it can be shown (provided the stopping conditions are never invoked and $h^{(k)}$ tends to zero as k tends to infinity) that the sequence $\{\breve{x}^{(j)}\}$

of QMPs is infinite and the cluster points of $\{\breve{x}^{(j)}\}$ are stationary points of f [22]. Furthermore as the sequence of function values $\{f(x^{(k)})\}$ decreases monotonically all cluster points have the same function value. Note that although algorithm Templates G1 and F1 are similar the points selected by Template G1 must belong to a grid and new grids can only be created once a GLM is found; but a new frame can be selected at *each* iteration of Template F1 and each frame is independent of any underlying grid. This gives the frame-based algorithm of Template F1 much greater flexibility than the grid-based algorithm of Template G1.

Arbitrary finite processes can usefully be included in Step 3 and either (or both) branches of Step 5 for Template G1 and Step 5 and either (or both) branches of Step 7 for Template F1 [21, 22]. Such processes could investigate the value of the objective function at finitely many points, implement a quasi-Newton step, perform a line search which is guaranteed to terminate or perhaps some other finite algorithm or heuristic that is not formulated in terms of grids or frames [21, 22]. Some of these ideas will be discussed in more detail in Chapters 3 and 4. A disadvantage of grid-based methods is that steps to arbitrary lower points can only occur when a GLM has been found [22] and this may be infrequent [66].

Opportunistic algorithms

The algorithms of Templates G1 and F1 require that the objective function is evaluated at all of the points which form a positive basis about the current iterate before the next iterate can be determined. If the objective function is expensive to evaluate it may be beneficial to abandon such a search as soon as a sufficient reduction in the objective function is obtained. Algorithms which are able to abandon a partially completed search in this manner are called *opportunistic* [65].

For frame-based methods opportunism means that the algorithm can abandon a partially completed frame immediately after locating a point of sufficient descent. Opportunistic versions of Templates G1 and F1 that are able to move to a new iterate whenever a point with sufficiently low function value is found are possible (see [21, 22] for more details). The price paid for opportunism is that the convergence theory applies only to the sequence of GLMs or ϵ-QMPs rather than to the entire sequence of iterates. To date there has been an unavoidable trade-off between these features [66]. This will be expanded upon in the next chapter.

Chapter 3
Frame-based algorithms for unconstrained optimisation

In this chapter the class of simple descent frame-based line search algorithms for unconstrained optimisation developed in [65] is presented and a new convergence proof is developed. Using an approach similar to that in [65] a provably convergent sufficient descent frame-based line search algorithm is also developed.

The convergence proof for the simple descent algorithm in [65] (in fact nearly all convergence proofs for frame-based algorithms) relies on the existence of a convergent subsequence of ordered positive bases whose limit is also an ordered positive basis. This requirement is usually enforced by requiring the existence of a subsequence of either structurally equivalent ordered positive bases or ordered positive bases of the same cardinality and construction. For example the following three methods are suggested in [65]:

(a) All positive bases $\mathcal{V}_+^{(k)} = \{v_i^{(k)}\}$ have cardinality $2n$ and $v_i^{(k)} = -v_{n+i}^{(k)}$ for each $i \in \{1, \ldots, n\}$ and for each $k \in \mathbb{N}$.

(b) The sequence of ordered positive bases is the union of a finite number of structurally equivalent subsets.

(c) All positive bases $\mathcal{V}_+^{(k)}$ have cardinality $n+1$ and there are negative constants τ_{SC_1} and τ_{SC_2} which bound the structure coefficients so that $\tau_{\text{SC}_1} \leqslant \zeta_{i,n+1}^{(k)} \leqslant \tau_{\text{SC}_2}$ for each $i \in \{1, \ldots, n\}$ and for all $k \in \mathbb{N}$.

Frame-based algorithms for unconstrained optimisation have tended to use one or a combination of minimal positive bases with $n+1$ elements, maximal positive bases with $2n$ elements or structurally equivalent positive bases. Such restrictions were originally imposed as a convenient way of constructing positive bases with the appropriate properties. Using ordered positive bases for example enables the construction of new positive bases by simply changing the underlying basis and also provides a convenient way of defining the limit of a sequence of positive bases. However the power of positive bases, which underlies all frame-based methods, is that the result of Theorem 2.1 (on page 16) holds for all positive bases not just certain subsets with special structural properties.

A more general theoretical approach that ensures the appropriate limits exist and are themselves positive bases assumes that each subsequence of ordered positive bases with cardinality m is contained in a compact subset $\mathcal{X}_m$ of the set of all ordered positive bases with cardinality m. This ensures that for each subsequence of ordered positive bases with cardinality m there exists a convergent sub-subsequence and since the sub-subsequence is contained in the compact set $\mathcal{X}_m$ its limit is also in $\mathcal{X}_m$ so it is an ordered positive basis with cardinality m [65]. In other words this guarantees that the cluster points of subsequences of ordered positive bases with cardinality m are also ordered positive bases with cardinality m. In more practical terms this theoretical approach is equivalent to placing the appropriate uniform bounds on the members of the positive bases and this is the approach taken in this thesis. No other requirements, such as ordering or structural equivalence, are necessary for the convergence proofs of the frame-based algorithms presented in this chapter.

The simple descent frame-based algorithm requires that, before a search direction is determined, the objective function is evaluated at a group of points including those which belong to a frame about the current iterate. Once the function has been evaluated at these points then, regardless of the size of the descent, any descent at least as good as that obtained from the best frame point is acceptable. Since at least $n+1$ function evaluations (one at each of the frame points) must be carried out before a search direction is determined it may be possible to build up useful finite difference derivative information along the way. Note that a method of generating finite difference gradient approximations with a built-in check on their accuracy is described in [23] where a least squares approximation of the gradient is compared to the finite difference approximation. A decision to switch between forward and central differences is made depending on how closely these approximations agree. This approach could be generalised for frame-based methods since for each frame at least $n+1$ directional derivative difference estimates are obtained and so there is always an overdetermined system for approximating the gradient.

Also in this chapter the simple descent frame-based line search algorithm from [65] is developed into an opportunistic sufficient descent frame-based line search algorithm for unconstrained optimisation. The opportunistic algorithm has the advantage that a new search direction can be determined as soon as sufficient progress is made. If, at the start of each iteration, a quasi-Newton or similar step is attempted that reduces the objective function by a sufficient amount then a new iterate can be found immediately without the need to complete a frame about the current iterate. Depending on the size of the positive basis this could save up to, between $n+1$ and $2n$, possibly expensive

function evaluations per iteration. In the worst case, when a frame must be completed about the current iterate before a search direction is determined, the sufficient descent and simple descent algorithms require the same number of function evaluations per iteration.

In the following chapter both the simple descent and sufficient descent algorithms for unconstrained optimisation are extended to handle linear constraints. Full convergence results are established for all four types of algorithm. In the next two sections the simple descent and sufficient descent frame-based line search algorithms for unconstrained optimisation are presented, along with their convergence proofs.

3.1 Simple descent frame-based line search algorithms

At each iteration of the simple descent frame-based line search algorithm for unconstrained optimisation developed in [65] the objective function f is evaluated at a finite number of points including those which form a frame around the current iterate $x^{(k)}$. These points may collectively be expressed in the form $x^{(k)} + h^{(k)}s$, where s ranges over the members of a finite set $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$. A member of $\mathcal{S}_+^{(k)}$, which is required to satisfy the adequate decrease condition

$$f(x^{(k)} + h^{(k)}s^{(k)}) \leqslant f(x^{(k)} + h^{(k)}v), \quad \forall v \in \mathcal{V}_+^{(k)}, \tag{3.1}$$

is then chosen as the search direction $s^{(k)}$. The step $s^{(k)}$ is known as the adequate descent step and the quantity

$$\Delta^{(k)} = f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)}) \tag{3.2}$$

measures the decrease in f due to the adequate descent step from $x^{(k)}$. If $\Delta^{(k)} \leqslant 0$ then the frame is minimal, the current iterate is relabelled $z^{(k)}$ and the frame size is reduced. Otherwise $\Delta^{(k)} > 0$ and a line search is conducted along the line $x^{(k)} + \alpha h^{(k)}s^{(k)}$ ($\alpha > 0$) and the point $z^{(k)} = x^{(k)} + \alpha^{(k)}h^{(k)}s^{(k)}$ is returned as the line search's outcome. There are many possible strategies for determining the point returned by the line search including analogues of the popular Goldstein-Armijo, one sided Wolfe-Powell and Armijo conditions. The algorithm may take any point not higher than $z^{(k)}$ as the next iterate $x^{(k+1)}$. This completes an iteration. The algorithm halts when the stopping conditions are satisfied.

Template U1: Unconstrained simple descent algorithm

Step 1: Set $k = 0$ and choose the initial point $x^{(0)}$.

Step 2: Choose $h^{(k)}$, $\mathcal{V}_+^{(k)}$ and $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$.

Step 3: Determine a search direction $s^{(k)}$.

Step 4: Set $\Delta^{(k)} = f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)})$.

Step 5: If $\Delta^{(k)} > 0$ then:

Perform a line search and let $z^{(k)}$ be the point selected by that search.

Otherwise:

Set $z^{(k)} = x^{(k)}$.

Step 6: Calculate f at a finite number of points and choose $x^{(k+1)}$ to be the lowest of these points and $z^{(k)}$.

Step 7: If the stopping conditions are satisfied then stop, otherwise increment k and go to Step 2.

Convergence of the algorithm

The algorithm described in Template U1 can produce two different types of iterates: those which are minimal frame centres ($\Delta^{(k)} \leqslant 0$) and those from which a line search is conducted. The former can be viewed as discrete approximations to minimisers whereas a descent step is known in advance for the latter. Template U1 may produce either (or both) types of iterates. If the initial point is the global minimiser of a function for example then only iterates which are minimal frame centres are produced. On the other hand appropriate choices of $\mathcal{S}_+^{(k)}$ and $h^{(k)}$ may result in few or no iterates being minimal frame centres.

Each $\mathcal{S}_+^{(k)}$ is a positive spanning set for $\mathbb{R}^n$ since it contains the positive basis $\mathcal{V}_+^{(k)}$ as a subset. Each $\mathcal{S}_+^{(k)}$ must contain only finitely many elements to ensure that Step 3 is a finite process. The choice of vectors which are added to $\mathcal{V}_+^{(k)}$ to form $\mathcal{S}_+^{(k)}$ is arbitrary apart from restrictions on length. This is one place where a quasi-Newton step could easily be included. Alternatively $\mathcal{V}_+^{(k)}$ could be chosen to include a quasi-Newton or other desired step.

The choice of the finite set of points in Step 6 is also arbitrary but any such point cannot be accepted as the next iterate unless it is better than the point $z^{(k)}$ chosen in Step 5. If no extra function values are calculated in Step 6 then a pair of identical iterates is generated whenever a minimal frame occurs.

Line searches. Convergence for the full sequence of iterates generated by algorithm Template U1 can be shown if each line search rejects steps that yield insufficient descent through either being too short or too long. The frame-based approach requires that $h^{(k)}$ tends to zero as k tends to infinity which together with an upper bound on $\|s^{(k)}\|$ ensures that an infinite sequence of steps that are too long is avoided. Hence Template U1 is ideally suited to forward tracking line searches.

A forward tracking line search generates an increasing sequence of values $\{\alpha_j\}$ for which the corresponding function values $f_j = f(x^{(k)} + \alpha_j h^{(k)} s^{(k)})$ are calculated until a pre-selected line search termination condition is satisfied. The first member in the sequence of α-values is $\alpha_1 = 1$ as this corresponds to the step $x^{(k)} + h^{(k)} s^{(k)}$. Each subsequent α-value must satisfy

$$\varrho_1 \alpha_{j-1} \leqslant \alpha_j \leqslant \varrho_2 \alpha_{j-1} \tag{3.3}$$

where $1 < \varrho_1 < \varrho_2$. For later convenience define $\alpha_0 = 0$ so that $f_0 = f(x^{(k)})$. Only the line search with termination criterion:

L1. $f_j \geqslant f_{j-1}$

is considered here (see [65] for five other line search possibilities). The penultimate value α_{j-1} is accepted as $\alpha^{(k)}$ and the point $z^{(k)} = x^{(k)} + \alpha^{(k)} h^{(k)} s^{(k)}$ is returned as the line search's outcome. Condition L1 ensures that a line local minimum lies in the α-interval $[\alpha_{j-2}, \alpha_j]$. Furthermore the actual function values are not required: only their relative ordering is important.

Assumptions (Simple descent algorithm). The following assumptions are used to prove the convergence of the simple descent algorithm described in Template U1:

A1. $f : \mathbb{R}^n \to \mathbb{R}$ is a $\mathcal{C}^1$ objective function.

A2. The sequence of all iterates $\{x^{(k)}\}$ is contained in a compact subset of $\mathbb{R}^n$.

A3. $\{h^{(k)}\}$ is a sequence of frame size parameters such that $h^{(k)} \to 0$ as $k \to \infty$.

A4. $\{\mathcal{V}_+^{(k)}\}$ is a sequence of positive bases for $\mathbb{R}^n$.

A5. There exist positive constants ξ and Ξ such that $\xi \leqslant \|v\| \leqslant \Xi$ for each $v \in \mathcal{V}_+^{(k)}$ and for each $k \in \mathbb{N}$.

A6. There exists a positive constant τ_θ such that for each $k \in \mathbb{N}$ and any non-zero $g \in \mathbb{R}^n$

$$\max_{v \in \mathcal{V}_+^{(k)}} \left\{ \frac{-v^\mathsf{T} g}{\|v\| \cdot \|g\|} \right\} \geqslant \tau_\theta > 0.$$

A7. $0 < \|s^{(k)}\| \leqslant \Xi$ for each $k \in \mathbb{N}$.

A8. The line search complies with Condition L1.

Assumption A3 ensures that an upper bound $h_{\max}$ exists for all members of the sequence $\{h^{(k)}\}$. Assumption A6 prevents the use of positive bases whose directions are all poor descent directions for the objective function whenever the gradient of the objective function is non-zero. This is achieved by ensuring that the maximum value of the direction cosines between $-g$ and members of the positive basis is positive and uniformly bounded away from zero. The importance of this uniform bound is illustrated in two dimensions in the following example [19].

Example 3.1. Let $f(x, y) = (x-1)^2 + y^2$ so that f has a minimum of zero at the minimiser $x^* = (1, 0)^\mathsf{T}$. Let $\{\theta^{(k)}\}$ be the sequence of angles defined by $\theta^{(k)} = \pi/2 - \sigma^{-k}$ for each $k \in \mathbb{N}$ and some $\sigma > 1$ so that $\theta^{(k)} \to \pi/2$ as $k \to \infty$. Let $\{\mathcal{V}_+^{(k)}\}$ be the sequence of positive bases for $\mathbb{R}^2$ defined by $\mathcal{V}_+^{(k)} = \{v_1^{(k)}, v_2^{(k)}, v_3^{(k)}\}$ where

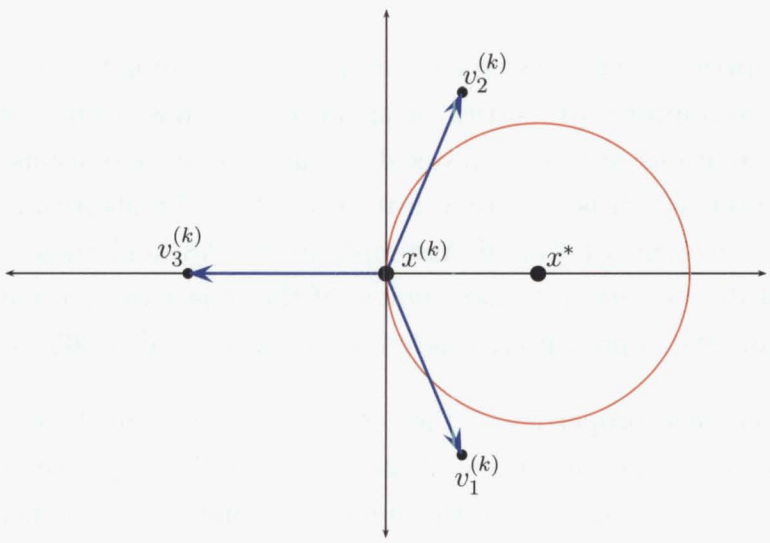

Figure 3.1: A circular contour, minimal frame and the importance of a uniform bound on the angle between positive basis vectors.

$v_1^{(k)} = (\cos\theta^{(k)}, -\sin\theta^{(k)})^\mathsf{T}$, $v_2^{(k)} = (\cos\theta^{(k)}, \sin\theta^{(k)})^\mathsf{T}$ and $v_3^{(k)} = (-1, 0)^\mathsf{T}$. Let the frame size parameter $h^{(k)} = 2\rho\cos\theta^{(k)}$ for each $k \in \mathbb{N}$ and some $\rho > 1$ so that $h^{(k)} \to 0$ as $k \to \infty$. Note that $2\cos\theta^{(k)}$ is the distance from the origin to a point on the unit circle centred on $(1, 0)^\mathsf{T}$ along the line at angle $\theta^{(k)}$ to the positive x-axis. Finally let $x^{(0)} = (0, 0)^\mathsf{T}$. Since $\det([v_1^{(k)}, v_2^{(k)}]) = \sin(2\theta^{(k)}) > 0$ for all $k \in \mathbb{N}$, $\mathcal{V}_+^{(k)}$ is a positive basis for each $k \in \mathbb{N}$. Furthermore the choice of $h^{(k)}$ ensures that each frame centred on $x^{(k)}$ is minimal for each $k \in \mathbb{N}$ and so the sequence $\{x^{(k)}\}$ converges to a non-stationary point of f. The situation is illustrated in Figure 3.1 which shows a single circular contour with height $f = 1$. As k increases the frame size decreases so that if the directions of the vectors $v_i^{(k)}$ were kept constant a point inside the contour circle (and hence with lower function value) would eventually be found. However the reduction of $h^{(k)}$ is offset by the increasing angle between the vectors $v_1^{(k)}$ and $v_2^{(k)}$ so that the frame points are always outside the circular contour shown in Figure 3.1. Despite the fact that each $\mathcal{V}_+^{(k)}$ is a positive basis for $\mathbb{R}^2$ and $h^{(k)} \to 0$ as $k \to \infty$ the algorithm converges to the non-stationary point $(0, 0)^\mathsf{T}$. The reason is that $\mathcal{V}_+^{(k)} \to \{(0, -1)^\mathsf{T}, (0, 1)^\mathsf{T}, (-1, 0)^\mathsf{T}\}$ as $k \to \infty$: which is not a positive basis for $\mathbb{R}^2$. This type of degenerate behaviour is prevented if the direction cosine between the negative gradient and at least one of the positive basis vectors is positive and bounded away from zero. Since the gradient may not be known orientating the positive basis is difficult unless there is a positive lower bound on the maximum direction cosine between *any* non-zero vector in $\mathbb{R}^n$ and the positive basis directions.

Convergence proof. The convergence of algorithm Template U1 is determined by examining the asymptotic properties of an arbitrary subsequence of iterates when the stopping conditions are never invoked. Practical considerations make stopping conditions essential which is why they feature in all of the algorithm templates [66]. It is shown in Theorem 3.1 that if Assumptions A1–A8 hold then cluster points of the sequence of iterates are stationary points of the objective function. Although the result is the same the approach used here is different to that in [65, p. 369]:

> The convergence properties of the template are examined by first fixing attention on a single arbitrary cluster point of the sequence of iterates. ... This cluster point can be the limit of an infinite subsequence either of minimal frame centres, or of iterates at which descent steps are known. These two types of subsequences must be handled separately. ... Here, only subsequences consisting entirely of minimal frames ($\Delta \leqslant 0$) or entirely of non-minimal frame centres ($\Delta > 0$) are considered.

The proof presented here allows the use of an arbitrary subsequence of iterates which converge to a cluster point. It is not necessary to consider the two types of iterates separately. The proof (by contradiction) of Theorem 3.1 proceeds by showing that if the gradient at the cluster point is non-zero then only a finite number of minimal frame centres are possible. The cluster point must therefore be the limit of an infinite sequence of non-minimal frame centres. If the gradient at such a cluster point is non-zero then the difference between function values of distinct frame centres is uniformly bounded away from zero preventing the convergence of the corresponding sequence of function values. This gives rise to a contradiction. Lemmas 3.1–3.4 and Corollary 3.2 provide useful intermediate results for Theorem 3.1. Lemma 3.1 shows that if the gradient at a cluster point is non-zero then the directional derivative of the objective function in at least one of the positive basis directions is negative and uniformly bounded away from zero as k tends to infinity.

Lemma 3.1. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ *is a sequence of iterates such that* $y^{(k)} \to y^{(\infty)}$ *as* $k \to \infty$,

(c) $\nabla f(y^{(\infty)}) \neq 0$,

there exists $K \in \mathbb{N}$, $\Theta > 0$ *and some* $v^{(k)} \in \mathcal{V}_+^{(k)}$ *such that*

$$v^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Theta, \quad \forall k \geqslant K.$$

Proof. By Assumption A5, $\|v^{(k)}\| \geqslant \xi$ for each $v^{(k)} \in \mathcal{V}_+^{(k)}$ and for each $k \in \mathbb{N}$. By Assumption A6, for each $k \in \mathbb{N}$ there is a minimum value, $\cos\theta^{(k)} \geqslant \tau_\theta$ for the largest direction cosine between $-\nabla f(y^{(k)})$ and the members of $\mathcal{V}_+^{(k)}$. Therefore

$$v^{(k)\mathsf{T}} \nabla f(y^{(k)}) \leqslant -\tau_\theta \xi \|\nabla f(y^{(k)})\|,$$

for some $v^{(k)} \in \mathcal{V}_+^{(k)}$. By the $\mathcal{C}^1$ continuity of f, $\nabla f(y^{(k)}) \to \nabla f(y^{(\infty)})$ as $k \to \infty$, so that for any $\beta \in (0,1)$ there exists $K \in \mathbb{N}$ such that $\|\nabla f(y^{(k)})\| > \beta \|\nabla f(y^{(\infty)})\|$, whenever $k \geqslant K$. Hence, for some $v^{(k)} \in \mathcal{V}_+^{(k)}$,

$$v^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\beta\tau_\theta \xi \|\nabla f(y^{(\infty)})\|, \quad \forall k \geqslant K. \qquad \square$$

The following corollary (which follows immediately from Lemma 3.1) shows that if the gradient at a cluster point is non-zero then the directional derivative of the objective function in the search direction is uniformly bounded away from zero as k tends to infinity. The existence of a search direction is ensured by only considering sequences of non-minimal frame centres.

Corollary 3.2. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ *is a sequence of non-minimal frame centres such that* $y^{(k)} \to y^{(\infty)}$ *as* $k \to \infty$,

(c) $\nabla f(y^{(\infty)}) \neq 0$,

there exists $K \in \mathbb{N}$ *and* $\Phi > 0$ *such that*

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Phi, \quad \forall k \geqslant K.$$

Proof. A Taylor series expansion of both sides of the adequate descent condition of equation (3.1) (on page 24) gives

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)}) \leqslant v^{\mathsf{T}} \nabla f(y^{(k)}) + o(1), \quad \forall v \in \mathcal{V}_+^{(k)} \text{ and } \forall k \in \mathbb{N}.$$

So by Lemma 3.1 there exists $K \in \mathbb{N}$ and $\Phi > 0$ such that

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Phi, \quad \forall k \geqslant K. \qquad \square$$

Lemma 3.3 shows that if the gradient at a cluster point is non-zero then the line search along the search direction $s^{(k)}$ eventually takes steps which are neither too short nor too long.

Lemma 3.3. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ *is a sequence of non-minimal frame centres such that* $y^{(k)} \to y^{(\infty)}$ *as* $k \to \infty$,

(c) $\nabla f(y^{(\infty)}) \neq 0$,

there exists $K \in \mathbb{N}$, $\tau_{LS_1} > 0$ *and* $\tau_{LS_2} > \tau_{LS_1}$ *such that* $\alpha^{(k)} h^{(k)} \in [\tau_{LS_1}, \tau_{LS_2}]$ *whenever* $k \geqslant K$.

Proof. For each $k \in \mathbb{N}$, line search Condition L1 ensures that a line local minimum lies in the interval $[\alpha_{j-2} h^{(k)}, \alpha_j h^{(k)}]$. Let $l_*^{(k)}$ be the location of such a line local minimum. For each $k \in \mathbb{N}$ there exists $J^{(k)} \in \mathbb{N}_0$ such that $\alpha_j h^{(k)} \leqslant l_*^{(k)}$ if and only if $j \leqslant J^{(k)}$. That is, $\alpha_{J^{(k)}} h^{(k)}$ is the line search step immediately on or before $l_*^{(k)}$ and $\alpha_{J^{(k)}+1} h^{(k)}$

(the very next step) is the line search step immediately after $l_*^{(k)}$. Let $P^{(k)} = \alpha_{J(k)} h^{(k)}$ and $Q^{(k)} = \alpha_{J(k)+1} h^{(k)}$ so that

$$P^{(k)} \leqslant l_*^{(k)} < Q^{(k)} \tag{3.4}$$

and

$$\alpha^{(k)} h^{(k)} \in \{P^{(k)}, Q^{(k)}\}. \tag{3.5}$$

By equation (3.3) (on page 26),

$$\varrho_1 P^{(k)} \leqslant Q^{(k)} \leqslant \varrho_2 P^{(k)}. \tag{3.6}$$

Rearranging equation (3.6) and combining with equation (3.4) gives

$$\frac{l_*^{(k)}}{\varrho_2} < \frac{Q^{(k)}}{\varrho_2} \leqslant P^{(k)} \qquad \text{and} \qquad Q^{(k)} \leqslant \varrho_2 P^{(k)} \leqslant \varrho_2 l_*^{(k)} \tag{3.7}$$

so that by equations (3.5) and (3.7)

$$\alpha^{(k)} h^{(k)} \in \left[\frac{l_*^{(k)}}{\varrho_2}, \varrho_2 l_*^{(k)} \right].$$

The line local minimiser $l_*^{(k)}$ must satisfy the first order necessary condition $s^{(k)\mathsf{T}} \nabla f(y^{(k)} + l_*^{(k)} s^{(k)}) = 0$. By Corollary 3.2 there exists $K \in \mathbb{N}$ such that $s^{(k)\mathsf{T}} \nabla f(y^{(k)})$ is uniformly bounded away from zero whenever $k \geqslant K$. Hence by the $\mathcal{C}^1$ continuity of f, $l_*^{(k)}$ is also uniformly bounded away from zero whenever $k \geqslant K$. Therefore there exists $\tau_{\text{LS}_1} > 0$ and, by Assumption A2 (which places an upper limit on the maximum step length), $\tau_{\text{LS}_2} > \tau_{\text{LS}_1}$ such that

$$\alpha^{(k)} h^{(k)} \in \left[\tau_{\text{LS}_1}, \tau_{\text{LS}_2}\right], \quad \forall k \geqslant K. \qquad \square$$

Lemma 3.4 shows that if the directional derivatives of the objective function in the search directions are negative and uniformly bounded away from zero then the differences $f(y^{(k+1)}) - f(y^{(k)})$ of the sequence of function values are eventually uniformly bounded away from zero.

Lemma 3.4. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ *is a sequence of non-minimal frame centres such that* $y^{(k)} \to y^{(\infty)}$ *as* $k \to \infty$,

(c) *there exists* $K \in \mathbb{N}$ *and* $\Phi > 0$ *such that* $s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Phi$, $\forall k \geqslant K$,

there exists $L \geqslant K$ *such that* $f(y^{(k+1)}) - f(y^{(k)})$ *is uniformly bounded away from zero whenever* $k \geqslant L$.

Proof. By the uniform continuity of ∇f there exists a Lipschitz constant $\mathfrak{L} > 0$ such that

$$\|\nabla f(y^{(k)} + \delta^{(k)} s^{(k)}) - \nabla f(y^{(k)})\| < \mathfrak{L}\, \|s^{(k)}\| \cdot |\delta^{(k)}|, \quad \forall k \in \mathbb{N}.$$

By Assumption A7, $\|s^{(k)}\| \leqslant \Xi$ for all $k \in \mathbb{N}$ so that

$$\left| s^{(k)\mathsf{T}} \left[\nabla f(y^{(k)} + \delta^{(k)} s^{(k)}) - \nabla f(y^{(k)}) \right] \right| < \mathfrak{L}\, \Xi^2 |\delta^{(k)}|, \quad \forall k \in \mathbb{N}.$$

Since $s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Phi$ whenever $k \geqslant K$ it follows that for any $\beta \in (0, 1)$

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)} + \delta^{(k)} s^{(k)}) < -\beta\Phi, \quad \forall k \geqslant K$$

whenever

$$\delta^{(k)} \in \left(0,\ \frac{(1-\beta)\Phi}{\mathfrak{L}\Xi^2}\right).$$

By Lemma 3.3 there exists $L \geqslant K$ and $\tau_{\text{LS}_1} > 0$ such that $\alpha^{(k)} h^{(k)} \geqslant \tau_{\text{LS}_1}$ whenever $k \geqslant L$. Line search Condition L1 ensures that the line search only proceeds while the sequence of function values it generates decreases monotonically. This, together with the continuity of f and the monotonicity of $\{f(y^{(k)})\}$, ensures that there exists

$$\delta \in \left(0,\ \min\left\{\tau_{\text{LS}_1}, \frac{(1-\beta)\Phi}{\mathfrak{L}\Xi^2}\right\}\right)$$

(independent of k) such that

$$f(y^{(k+1)}) \leqslant f(y^{(k)} + \alpha^{(k)} h^{(k)} s^{(k)}) < f(y^{(k)} + \delta s^{(k)}) < f(y^{(k)})$$

whenever $k \geqslant L$. Hence

$$\begin{aligned} f(y^{(k+1)}) - f(y^{(k)}) &< f(y^{(k)} + \delta s^{(k)}) - f(y^{(k)}) \\ &= \int_0^{\delta} s^{(k)\mathsf{T}} \nabla f(y^{(k)} + \vartheta s^{(k)})\, \mathrm{d}\vartheta \\ &< -\delta\beta\Phi, \quad \forall k \geqslant L. \end{aligned}$$

□

The convergence results for the sequence of iterates generated by algorithm Template U1 are now presented.

Theorem 3.1. *Given Assumptions A1–A8 (on page 26) hold, cluster points of the sequence of iterates generated by algorithm Template U1 are stationary points of the objective function.*

Proof. Let $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ be any subsequence of iterates such that $y^{(k)} \to y^{(\infty)}$ as $k \to \infty$. Suppose $\nabla f(y^{(\infty)}) \neq 0$. By Lemma 3.1 there exists $K \in \mathbb{N}$ and $\Theta > 0$ such that for all $k \geqslant K$ there exists $v^{(k)} \in \mathcal{V}_+^{(k)}$ such that $v^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Theta$. By Taylor series

$$f(y^{(k)} + h^{(k)} v^{(k)}) < f(y^{(k)}) - \Theta\, h^{(k)} + o(h^{(k)}), \quad \forall k \geqslant K.$$

Since $h^{(k)} \to 0$ as $k \to \infty$ there exists $L \geqslant K$ such that $f(y^{(k)} + h^{(k)} v^{(k)}) < f(y^{(k)})$ whenever $k \geqslant L$. Hence the sequence $\{y^{(k)}\}_{k \geqslant L}$ consists entirely of non-minimal frame centres. By Corollary 3.2 there exists $M \geqslant L$ and $\Phi > 0$ such that $s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Phi$ whenever $k \geqslant M$. By Lemma 3.4 there exists $N \geqslant M$ such that $f(y^{(k+1)}) - f(y^{(k)})$ is uniformly bounded away from zero whenever $k \geqslant N$ so that $f(y^{(k)}) \not\to f(y^{(\infty)})$ as $k \to \infty$, a contradiction of the $\mathcal{C}^1$ continuity of f. Hence $\nabla f(y^{(\infty)}) = 0$. □

This completes the convergence proof. Cluster points of the full sequence of iterates generated by the simple descent frame-based line search algorithm for unconstrained optimisation presented in Template U1 are stationary points of $\mathcal{C}^1$ objective functions under mild conditions. Furthermore as the objective function decreases monotonically distinct cluster points have the same function value.

In the next section a sufficient descent frame-based algorithm is developed. Unlike the algorithm in Template U1 the sufficient descent algorithm allows opportunistic movement to a new iterate whenever the objective function is reduced by a sufficient amount.

3.2 Sufficient descent frame-based line search algorithms

The sufficient descent frame-based line search algorithm described in this section is based on the algorithms developed in [21, 65]. It differs from the simple descent algorithm of the previous section in that opportunistic movement is possible whenever a search direction is found that reduces the objective function by a sufficient amount.

At each iteration candidate search directions from a finite set $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$ are examined. The set of search directions $\mathcal{S}_+^{(k)}$ contains the positive basis $\mathcal{V}_+^{(k)}$ as a subset so that either a suitable search direction is found or a frame is completed about the current iterate. A search direction $s^{(k)} \in \mathcal{S}_+^{(k)}$ is accepted without the need to complete a frame about the current iterate whenever the sufficient descent condition

$$\Delta^{(k)} > \epsilon^{(k)} \tag{3.8}$$

is met. The quantity $\epsilon^{(k)}$ is known as the sufficient descent parameter. The quantity $\Delta^{(k)}$, introduced in equation (3.2) on page 24, measures the decrease in the objective function due to the step $h^{(k)}s^{(k)}$.

If $\Delta^{(k)} > \epsilon^{(k)}$ for some $s^{(k)} \in \mathcal{S}_+^{(k)}$ then a line search is conducted along $x^{(k)}+\alpha h^{(k)}s^{(k)}$ $(\alpha > 0)$ for a point $z^{(k)} = x^{(k)} + \alpha^{(k)}h^{(k)}s^{(k)}$. Otherwise $\Delta^{(k)} \leqslant \epsilon^{(k)}$ for every $s \in \mathcal{S}_+^{(k)}$ and so an $\epsilon^{(k)}$-quasi-minimal frame exists about the iterate $x^{(k)}$. In this situation the algorithm proceeds exactly like the simple descent algorithm of Template U1: if there is a search direction $s^{(k)}$ that satisfies the adequate decrease condition (on page 24) and $\Delta^{(k)} > 0$ then a line search is conducted along $x^{(k)} + \alpha h^{(k)}s^{(k)}$ $(\alpha > 0)$ for a point $z^{(k)} = x^{(k)} + \alpha^{(k)}h^{(k)}s^{(k)}$; otherwise the frame is minimal, the current iterate is relabelled $z^{(k)}$ and the frame size is reduced. The algorithm may take any point not higher than $z^{(k)}$ as the next iterate $x^{(k+1)}$. This completes an iteration. The algorithm halts when the stopping conditions are satisfied.

Template U2: Unconstrained sufficient descent algorithm

Step 1: Set $k = 0$ and choose the initial point $x^{(0)}$.

Step 2: Choose $h_{\min}^{(k)} > 0$ and $\epsilon_{\min}^{(k)} > 0$.

Step 3: Choose $h^{(k)} \geqslant h_{\min}^{(k)}$, $\epsilon^{(k)} \geqslant \epsilon_{\min}^{(k)}$, $\mathcal{V}_+^{(k)}$ and $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$.

Step 4: Calculate $\Delta^{(k)} = \max\{f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)})\}$ until $\Delta^{(k)} > \epsilon^{(k)}$ or every $s^{(k)} \in \mathcal{S}_+^{(k)}$ has been used.

Step 5: (a) If $\Delta^{(k)} > \epsilon^{(k)}$ then (a sufficient descent step exists):
Perform a line search and let $z^{(k)}$ be the point selected by that search.

(b) Otherwise (the frame is $\epsilon^{(k)}$-quasi-minimal):
If $\Delta^{(k)} > 0$ then:
Perform a line search and let $z^{(k)}$ be the point selected by that search.

Otherwise (the frame is minimal):
Set $z^{(k)} = x^{(k)}$.

Step 6: Calculate f at a finite number of points and choose $x^{(k+1)}$ to be the lowest of these points and $z^{(k)}$.

Step 7: If the stopping conditions are satisfied then stop.

Step 8: If $\Delta^{(k)} > \epsilon^{(k)}$ then:
Set $h_{\min}^{(k+1)} = h_{\min}^{(k)}$ and $\epsilon_{\min}^{(k+1)} = \epsilon_{\min}^{(k)}$. Increment k and go to Step 3.
Otherwise:
Increment k and go to Step 2.

Each $\mathcal{S}_+^{(k)}$ must contain only finitely many elements in order to ensure that Step 4 is a finite process. The choice of vectors which are added to $\mathcal{V}_+^{(k)}$ to form $\mathcal{S}_+^{(k)}$ is arbitrary apart from restrictions on length. As with the simple descent algorithm this is one place where a quasi-Newton step could easily be included.

The choice of the finite set of points in Step 6 is also arbitrary but any such point cannot be accepted as the next iterate unless it is better than the point $z^{(k)}$ chosen in Step 5. If no extra function values are calculated in Step 6 then a pair of identical iterates is generated whenever a minimal frame occurs.

Convergence of the algorithm

The algorithm described in Template U2 can be viewed as two nested loops. The outer loop selects positive values for the minimum frame size $h_{\min}^{(k)}$ and the minimum sufficient descent parameter $\epsilon_{\min}^{(k)}$. The inner loop locates an $\epsilon^{(k)}$-quasi-minimal frame centre while the values of $h_{\min}^{(k)}$ and $\epsilon_{\min}^{(k)}$ are fixed [21]. That is, the inner loop locates an $\epsilon^{(k)}$-quasi-minimal frame centre for an $\epsilon^{(k)}$ that is uniformly bounded away from zero. The outer loop generates a sequence of these quasi-minimal frame centres as $h^{(k)}$ tends to zero. Unlike the simple descent algorithm in Template U1, which completes a frame about the current iterate at every iteration, the sufficient descent algorithm is only guaranteed to complete a frame if a sufficient descent search direction is not found. It is only in this case that a new arbitrary frame size and sufficient descent parameter can be chosen. Until such time the frame size cannot be reduced below the lower bound $h_{\min}^{(k)}$ and the sufficient descent parameter cannot be reduced below the lower bound $\epsilon_{\min}^{(k)}$. Thus the sufficient descent parameter $\epsilon^{(k)}$ is uniformly bounded away from zero during the inner loop of Template U2. The $\mathcal{C}^1$ continuity of the objective function and the bounded nature of the iterates implies that only a finite number of sufficient

descent steps are possible before a quasi-minimal frame centre is located and so the inner loop of Template U2 is a finite process [21].

The convergence of Template U2 is established in Theorems 3.2–3.4. Theorems 3.2 and 3.3 where originally established in [21] and are included here with minor adaptations so that they apply to algorithm Template U2. Theorem 3.2 shows that the sequence of quasi-minimal frames is infinite.

Theorem 3.2. *Given Assumptions A1–A8 (on page 26) hold, algorithm Template U2 generates an infinite sequence of quasi-minimal frame centres* [21].

Proof. Let $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ be an arbitrary infinite subsequence of iterates. By Assumption A1 the objective function has C^1 continuity and by Assumption A2 the sequence of iterates $\{y^{(k)}\}$ is contained in a compact subset of $\mathbb{R}^n$ so that the sequence of function values $\{f(y^{(k)})\}$ is bounded below. Suppose the sequence $\{y^{(k)}\}$ contains only finitely many quasi-minimal frame centres. Then there exists $K \in \mathbb{N}$ such that the sequence $\{y^{(k)}\}_{k \geqslant K}$ does not contain any quasi-minimal frame centres. Therefore $f(y^{(k)}) - f(y^{(k+1)}) > \epsilon^{(k)} \geqslant \epsilon_{\min}^{(k)}$ for all $k \geqslant K$. Since the minimum sufficient descent parameter $\epsilon_{\min}^{(k)}$ can only be altered once a quasi-minimal frame centre is found $\epsilon_{\min}^{(k)} = \epsilon_{\min}^{(K)}$ for all $k \geqslant K$. Therefore the sequence of function values $\{f(y^{(k)})\}_{k \geqslant K}$ is unbounded below, a contradiction. Hence the sequence $\{y^{(k)}\}$ contains infinitely many quasi-minimal frame centres. □

The following definitions are useful for the remaining theorems in this chapter.

Definition 3.1 (Big-oh). A sequence $\{\epsilon^{(k)}\}$ is said to be big-oh of $h^{(k)}$ written as $\epsilon^{(k)} = \mathcal{O}(h^{(k)})$ if there exists a sequence $\{h^{(k)}\}$ and a positive constant τ_{ϵ_2} such that $|\epsilon^{(k)}/h^{(k)}| \leqslant \tau_{\epsilon_2}$ for all $k \in \mathbb{N}$.

Definition 3.2 (Little-oh). A sequence $\{\epsilon^{(k)}\}$ is said to be little-oh of $h^{(k)}$ written as $\epsilon^{(k)} = o(h^{(k)})$ if there exists a sequence $\{h^{(k)}\}$ such that $h^{(k)} \to 0$ as $k \to \infty$ and $\epsilon^{(k)}/h^{(k)} \to 0$ as $k \to \infty$.

Definition 3.3 (Strictly big-oh). A sequence $\{\epsilon^{(k)}\}$ is said to be strictly big-oh of $h^{(k)}$ written as $\epsilon^{(k)} = \bar{\mathcal{O}}(h^{(k)})$ if there exists a sequence $\{h^{(k)}\}$ and positive constants τ_{ϵ_1} and τ_{ϵ_2} such that $\tau_{\epsilon_1} \leqslant |\epsilon^{(k)}/h^{(k)}| \leqslant \tau_{\epsilon_2}$ for all $k \in \mathbb{N}$.

Definition 3.4 (Bounded little-oh). A sequence $\{\epsilon^{(k)}\}$ is said to be $g^{(k)}$-bounded little-oh of $h^{(k)}$ written as $\epsilon^{(k)} = \bar{o}_g(h^{(k)})$ if $\epsilon^{(k)} = o(h^{(k)})$ and there exists a sequence $\{g^{(k)}\}$ such that

$$\frac{|\epsilon^{(k)}|}{|h^{(k)}|} \geqslant |g^{(k)}|, \quad \forall k \in \mathbb{N}.$$

Theorem 3.3 shows that cluster points of the sequence of quasi-minimal frame centres generated by algorithm Template U2 are stationary points of $\mathcal{C}^1$ objective functions whenever the sufficient descent parameter $\epsilon^{(k)} = o(h^{(k)})$.

Theorem 3.3. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\epsilon^{(k)} = o(h^{(k)})$,

cluster points of the sequence of quasi-minimal frame centres generated by algorithm Template U2 are stationary points of the objective function [21].

Proof. Let $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ be an arbitrary infinite sequence of quasi-minimal frame centres so that

$$f(y^{(k)} + h^{(k)}v^{(k)}) \geqslant f(y^{(k)}) - \epsilon^{(k)}, \quad \forall v^{(k)} \in \mathcal{V}_+^{(k)} \text{ and } \forall k \in \mathbb{N}. \tag{3.9}$$

By a Taylor series expansion of equation (3.9)

$$v^{(k)\mathsf{T}}\nabla f(y^{(k)}) \geqslant o(1), \quad \forall v^{(k)} \in \mathcal{V}_+^{(k)} \text{ and } \forall k \in \mathbb{N}$$

so by Theorem 2.1 (on page 16), $\nabla f(y^{(k)}) \to 0$ as $k \to \infty$. □

Convergence results for the sequence of quasi-minimal frame centres for the particular sufficient descent function $\epsilon^{(k)} = N(h^{(k)})^\nu$ (for some $N > 0$ and $\nu > 1$) are developed in [21]. Convergence results for the sequence of quasi-minimal frame centres under the more general sufficient descent condition $\epsilon^{(k)} = o(h^{(k)})$ are developed in [66]. To date the results presented in Theorems 3.2 and 3.3 are the only convergence results for sufficient descent frame-based algorithms. As reported in [66, p. 418]:

> The price paid for opportunism is that the convergence theory applies only to the subsequence of quasi-minimal iterates. ... [Whereas] the convergence theory applies to the whole sequence of iterates [for non-opportunistic algorithms].

The effects of different sufficient descent parameters are now examined and the established convergence results of Theorems 3.2 and 3.3 are extended. It is shown in Theorem 3.4 that, for appropriate choices of the sufficient descent parameter, convergence results for the entire sequence of iterates can be obtained whilst maintaining the benefits of a sufficient descent algorithm.

Theorem 3.4. *Given:*

(a) *Assumptions A1–A8 (on page 26) hold,*

(b) $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ *is an arbitrary sequence of sufficient descent steps generated by algorithm Template U2,*

then:

(A) *Without further assumptions no information can be obtained about the gradient at cluster points of the sequence of sufficient descent steps generated by algorithm Template U2 for a general little-oh sufficient descent parameter.*

(B) *For a strictly big-oh sufficient descent parameter the sequence of sufficient descent steps generated by algorithm Template U2 is finite.*

(C) *There exists a bounded little-oh sufficient descent parameter such that the cluster points of the sequence of sufficient descent steps generated by algorithm Template U2 are stationary points of the objective function.*

Proof. By the definition of sufficient descent

$$f(y^{(k)} + h^{(k)}s^{(k)}) < f(y^{(k)}) - \epsilon^{(k)}, \quad \forall y^{(k)} \in \{y^{(k)}\}$$

and so by Taylor series

$$s^{(k)\mathsf{T}}\nabla f(y^{(k)}) < -\frac{\epsilon^{(k)}}{h^{(k)}} + o(1). \tag{3.10}$$

Either $\{y^{(k)}\}$ is finite so that no asymptotic convergence results are possible or $\{y^{(k)}\}$ is infinite with at least one cluster point. Let $\{y^{(k)}\}$ be an infinite sequence of sufficient descent iterates such that $y^{(k)} \to y^{(\infty)}$ as $k \to \infty$.

(A) Proof: Let $\epsilon^{(k)} = o(h^{(k)})$ so that by equation (3.10)

$$s^{(k)\mathsf{T}}\nabla f(y^{(k)}) < o(1).$$

If there exists $K \in \mathbb{N}$ and $\Phi > 0$ such that $s^{(k)\mathsf{T}}\nabla f(y^{(k)}) < -\Phi$ whenever $k \geqslant K$ then Lemma 3.4 implies that $f(y^{(k+1)}) - f(y^{(k)})$ is uniformly bounded away from zero so that $f(y^{(k)}) \not\to f(y^{(\infty)})$ as $k \to \infty$, a contradiction of the $\mathcal{C}^1$ continuity of f. Hence no such Φ exists so that $s^{(k)\mathsf{T}}\nabla f(y^{(k)}) \to 0$ as $k \to \infty$. Hence

$$|\cos\theta^{(k)}| \cdot \|\nabla f(y^{(k)})\| \to 0, \quad \text{as } k \to 0 \tag{3.11}$$

where $\cos\theta^{(k)}$ is the direction cosine between the sufficient descent search direction $s^{(k)}$ and the gradient $\nabla f(y^{(k)})$. Since the search direction $s^{(k)}$ may not be in the same direction as any of the positive basis directions satisfying Assumption A6 it is impossible to obtain any information about the gradient at a cluster point of the sequence of sufficient descent steps without further information on the values of the direction cosines: equation (3.11) is satisfied if either $\cos\theta^{(k)} \to 0$ as $k \to \infty$, or $\nabla f(y^{(k)}) \to 0$ as $k \to \infty$ (or both). □

(B) Proof: Let $\epsilon^{(k)} = \bar{O}(h^{(k)})$ so that (by definition) there exist positive constants τ_{ϵ_1} and τ_{ϵ_2} such that

$$\tau_{\epsilon_1} \leqslant \frac{\epsilon^{(k)}}{h^{(k)}} \leqslant \tau_{\epsilon_2}.$$

By equation (3.10)

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\tau_{\epsilon_1} + o(1)$$

and so by Lemma 3.4, $f(y^{(k+1)}) - f(y^{(k)})$ is bounded away from zero so that $f(y^{(k)}) \not\to f(y^{(\infty)})$ as $k \to \infty$, a contradiction of the $\mathcal{C}^1$ continuity f. Hence the sequence $\{y^{(k)}\}$ is finite. □

(C) Proof: Let $\epsilon^{(k)} \geqslant \Lambda\, h^{(k)}\, \|\nabla f(y^{(k)})\|$ for some $\Lambda > 0$. By equation (3.10)

$$s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\Lambda \|\nabla f(y^{(k)})\| + o(1).$$

By the $\mathcal{C}^1$ continuity of f, $\nabla f(y^{(k)}) \to \nabla f(y^{(\infty)})$ as $k \to \infty$. If $\nabla f(y^{(\infty)}) \neq 0$ then for any $\beta \in (0,1)$ there exists $K \in \mathbb{N}$ such that $s^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\beta\Lambda \|\nabla f(y^{(\infty)})\|$ whenever $k \geqslant K$ and so by Lemma 3.4, $f(y^{(k+1)}) - f(y^{(k)})$ is uniformly bounded away from zero whenever $k \geqslant K$. Hence, $f(y^{(k)}) \not\to f(y^{(\infty)})$ as $k \to \infty$, a contradiction of the $\mathcal{C}^1$ continuity of f. Hence $\nabla f(y^{(\infty)}) = 0$. □

The proof of item (A) in Theorem 3.4 explains why no convergence results have been established for sequences of sufficient descent steps for general $o(h^{(k)})$ sufficient descent parameters. If the sufficient descent parameter is "larger" than $o(h^{(k)})$ (that is, $\epsilon^{(k)} = \bar{O}(h^{(k)})$) then item (B) of Theorem 3.4 shows that the sequence of sufficient descent steps must be finite. Theorem 3.1 then shows that cluster points of the sequence of iterates generated by algorithm Template U2 are stationary points of the objective function since algorithm Template U2 behaves exactly like algorithm Template U1 whenever sufficient descent steps are not made. The proof of item (C) in Theorem 3.4 shows that if the sufficient descent parameter is $\|\nabla f(y^{(k)})\|$-bounded $o(h^{(k)})$ then cluster

points of the sequence of sufficient descent steps are stationary points of the objective function. Of course this approach requires the value of $\|\nabla f(y^{(k)})\|$ which may not be explicitly available. However whenever a quasi-minimal frame is completed about the current iterate an approximation to $\|\nabla f(y^{(k)})\|$ can be obtained. At a quasi-minimal frame centre at least one of the directions in the associated positive basis $\mathcal{V}_+^{(k)}$ is a descent direction for the objective function satisfying the bounds in Assumptions A1–A8 (on page 26) so that for each such $k \in \mathbb{N}$ there exists some $v^{(k)} \in \mathcal{V}_+^{(k)}$ such that

$$\tau_\theta \|v^{(k)}\| \cdot \|\nabla f(y^{(k)})\| \leqslant |v^{(k)\mathsf{T}} \nabla f(y^{(k)})| \leqslant \|v^{(k)}\| \cdot \|\nabla f(y^{(k)})\|$$

and so

$$\frac{|v^{(k)\mathsf{T}} \nabla f(y^{(k)})|}{\|v^{(k)}\|} \leqslant \|\nabla f(y^{(k)})\| \leqslant \frac{|v^{(k)\mathsf{T}} \nabla f(y^{(k)})|}{\tau_\theta \|v^{(k)}\|}.$$

Re-arranging a Taylor series expansion for $f(y^{(k)} + h^{(k)} v^{(k)})$ gives

$$v^{(k)\mathsf{T}} \nabla f(y^{(k)}) = \frac{1}{h^{(k)}} \left[f(y^{(k)} + h^{(k)} v^{(k)}) - f(y^{(k)}) \right] + o(1) \tag{3.12}$$

and so if the value of the sufficient descent parameter $\epsilon^{(k)}$ is too large then a quasi-minimal frame will be completed about the current iterate. A new approximation for $\|\nabla f(y^{(k)})\|$ can obtained and a more appropriate value of $\epsilon^{(k)}$ can be determined. Furthermore, by equation (3.12), the error in such an approximation tends to zero as k tends to infinity so that a self-correcting sequence of appropriately sized sufficient descent parameters could be determined automatically by the algorithm.

Chapter 4
Linear constraints

Following the approach developed in [66] the frame-based line search algorithms for unconstrained optimisation presented in Chapter 3 are now extended for linearly constrained optimisation. The general linearly constrained optimisation problem can be written as

$$\begin{aligned} \text{minimise} \quad & f(x), \\ \text{subject to} \quad & c_i(x) = 0, \quad \forall i \in \mathcal{E} \\ & c_i(x) \geqslant 0, \quad \forall i \in \mathcal{I} \end{aligned} \tag{4.1}$$

where $c_i(x) = a_i^{\mathsf{T}} x - b_i$ for the constant, non-zero constraint normals $a_i \in \mathbb{R}^n$ and the affine translations $b_i \in \mathbb{R}$. Although some authors refer to constraints of this type as affine constraints (whenever b_i is non-zero) they are called linear constraints throughout this thesis. The algorithms for linearly constrained optimisation developed in this chapter are based on the unconstrained algorithms developed in the previous chapter. If progress is unimpeded by any of the constraints then the unconstrained and linearly constrained algorithms are identical. However the positive bases of Chapter 3 are replaced with *aligned positive bases* of the feasible region whenever an iterate is "sufficiently close" to a constraint and the line search is replaced by a *feasible* line search. Aligned positive bases, feasible line searches and various ways of deciding when an iterate is close to a constraint are discussed in more detail in the following sections.

4.1 Aligned positive bases

A key feature of frame-based algorithms for unconstrained optimisation is that whenever the gradient of the objective function is non-zero at least one of the vectors in the associated positive basis is a descent direction for the objective function. To maintain this feature in linearly constrained optimisation it is necessary to create a positive basis whose subsets are able to positively span the feasible regions defined by sets of constraints which are active at (and near) the current iterate. Specifically, subsets of the positive basis must positively span every possible cone of feasible directions at (and near) the current iterate [66].

Figure 4.1 helps illustrate the importance of aligning the positive basis with nearby constraints. To ensure that at least one positive basis vector is a feasible descent

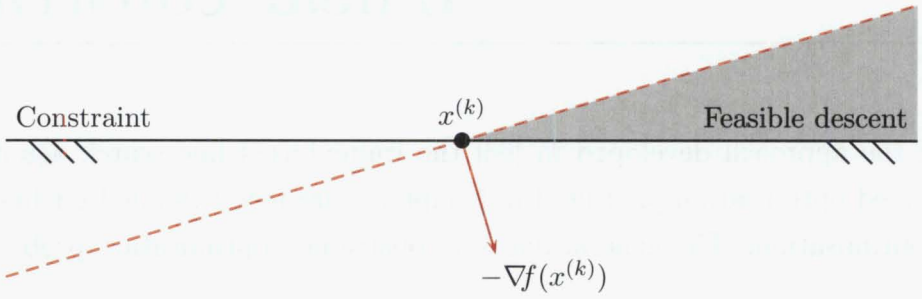

Figure 4.1: Feasible descent directions.

direction at least one positive basis vector must lie inside the feasible descent region (shaded grey in Figure 4.1). The authors of [48, p. 1084] note a similar observation in their article on pattern search with simple bounds:

> The problem is that while there are feasible directions of descent emanating from $x^{(k)}$, our pattern is not oriented in such a way as to capture any of this information from its feasible point .... A moment's reflection reveals that the problem is that the pattern does not allow us to move parallel to the bounds.

Figure 4.2 shows the same feasible region and a set of feasible directions which positively span the cone of feasible directions from the current iterate $x^{(k)}$. The authors of [49, p. 917–918] note in their follow-up paper on pattern search with linear constraints that:

> Key to the analysis of the algorithms is the way in which the local search patterns conform to the geometry of the boundary of the feasible region. ... The general idea, which also applies to unconstrained minimisation, is that the pattern must contain search directions that comprise a set of generators for the cone of feasible directions. ...we must also take into account the constraints that are almost binding in order to take sufficiently long steps.

At each iteration the constraints which are active at and sufficiently close to the current iterate are identified and a positive basis is constructed so that for any cone of feasible directions defined by some subset of these constraints there exists a subset of the frame's positive basis which positively spans the cone of feasible directions. Such positive bases are said to be aligned with the cone of feasible directions, or more

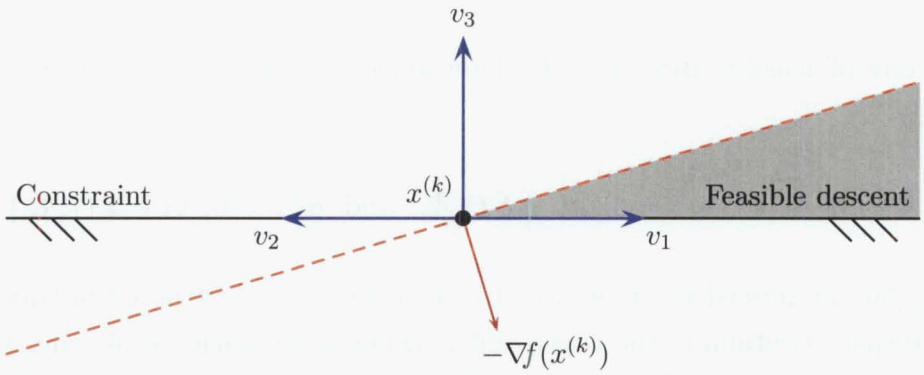

Figure 4.2: Spanning set aligned with the feasible region.

simply, aligned positive bases [66]. The corresponding frames are called aligned frames. Aligning the positive bases in this way guarantees that at least one positive basis vector will be a feasible descent direction (as illustrated in Figure 4.2) whenever a feasible descent region exists. The active constraint working set $\mathcal{A}_+(x) \supseteq \mathcal{A}(x)$ is the index set for the constraints which are active at (and near) the point x; this is a superset of $\mathcal{A}(x)$, the set of constraints which are active at x. The abbreviation $\mathcal{A}_+^{(k)}$ will be used for the working set $\mathcal{A}_+(x^{(k)})$. To maintain feasibility the current working set must include all of the equality constraints. Hence $\mathcal{A}_+^{(k)} \supseteq \mathcal{E}$ for every $k \in \mathbb{N}$. Two possibilities for determining when to include nearby inequality constraints in the current working set are:

(a) The constraint value at the current iterate $x^{(k)}$ is sufficiently small so that, for some (typically small) positive constant $\tau_{\mathcal{A}}$,

$$|a_i^{\mathsf{T}} x^{(k)} - b_i| < \tau_{\mathcal{A}} \implies i \in \mathcal{A}_+^{(k)} \quad [66].$$

(b) The (orthogonal) distance from the current iterate to a constraint is sufficiently small. This is expanded upon using the Euclidean norm later in this section.

Constructing aligned positive bases. To simplify the presentation it will be assumed that the constraints in each working set are linearly independent. This is not a necessary requirement for the convergence of the algorithms in this chapter. As noted in [66, pp. 430–431] when degeneracy is present the existence of an aligned positive spanning set of the feasible cone is still guaranteed [75] but its construction may be computationally expensive [49]. To further simplify the presentation, the use of iteration superscripts will be temporarily suspended throughout the remainder of this section.

The cone of feasible directions $\mathcal{K}$ determined by the current working set $\mathcal{A}_+$ is defined as

$$\mathcal{K} = \{z \in \mathbb{R}^n : a_i^\mathsf{T} z = 0,\ \forall i \in \mathcal{E} \cap \mathcal{A}_+ \text{ and } a_i^\mathsf{T} z \geqslant 0,\ \forall i \in \mathcal{I} \cap \mathcal{A}_+\}.$$

Following [66] an aligned positive basis for the cone $\mathcal{K}$ is constructed in two parts: (a) for the subspace containing the constraint normals a_i for each $i \in \mathcal{A}_+$ and (b) for the orthogonal complement of this subspace. Each of these parts is constructed as follows.

(a) *Constraint normal subspace:* Let A be the $n \times n$ invertible matrix whose first $|\mathcal{A}_+|$ columns are the (linearly independent) constraint normals for each of the constraints in the current working set and whose remaining $n - |\mathcal{A}_+|$ columns are chosen so that all n columns of A are linearly independent. Now let $U^\mathsf{T} = A^{-1}$ so that

$$U^\mathsf{T} a_{i_q} = e_q \tag{4.2}$$

where e_q denotes the qth Euclidean coordinate unit vector and each $i \in \mathcal{A}_+$ is indexed as i_q for $q \in \{1, \ldots, |\mathcal{A}_+|\}$. Hence, for each column u_p of U,

$$u_p^\mathsf{T} a_{i_q} = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{otherwise.} \end{cases}$$

This ensures that u_q points into the feasible region of constraint c_{i_q} (since $u_q^\mathsf{T} a_{i_q} > 0$) and that u_p is in the nullspace of a_{i_q} (since $u_p^\mathsf{T} a_{i_q} = 0$) whenever $p \neq q$.

(b) *Orthogonal complement subspace:* If $|\mathcal{A}_+| < n$ then construct a set of vectors $\mathcal{W}_+$ such that

$$(Uw)^\mathsf{T} a_{i_q} = 0, \quad \forall q \in \{1, \ldots, |\mathcal{A}_+|\} \text{ and } \forall w \in \mathcal{W}_+. \tag{4.3}$$

If $w \in \mathbb{R}^n$ satisfies equation (4.3) then by equation (4.2) the qth component of w is zero for each $q \in \{1, \ldots, |\mathcal{A}_+|\}$ and the remaining $n - |\mathcal{A}_+|$ components of w are arbitrary. Let $\mathcal{W}_+$ be any positive basis for the orthogonal complement of the space spanned by the vectors $\{e_q\}$. The set of vectors $\{Uw : w \in \mathcal{W}_+\}$ is a positive basis for the nullspace of A^T. If $|\mathcal{A}_+| = n$ then $\mathcal{W}_+$ is the empty set.

Combining parts (a) and (b)

$$\mathcal{V} = \{u_q : i_q \in \mathcal{I} \cap \mathcal{A}_+\} \cup \{Uw : w \in \mathcal{W}_+\}$$

is a minimal set of generators for the cone $\mathcal{K}$ [66]. The set

$$\mathcal{V}_+ = \{\pm u_q : i_q \in \mathcal{A}_+\} \cup \{Uw : w \in \mathcal{W}_+\}$$

is an aligned positive basis for $\mathbb{R}^n$ [66]. If the working set $\mathcal{A}_+$ is empty then the set $\{u_q\}$ is also empty so that $\mathcal{W}_+$ is any positive basis for $\mathbb{R}^n$; $\mathcal{V} = \mathcal{V}_+ = \{Uw : w \in \mathcal{W}_+\}$ and, for convenience, U can be chosen to be the identity matrix, although any other invertible matrix is also acceptable.

For later convenience the set $\mathfrak{E} = \{-u_q : i_q \in \mathcal{A}_+\}$ will be referred to as the set of *exterior* aligned positive basis directions and the set $\mathfrak{I} = \mathcal{V}_+ \backslash \mathfrak{E}$ will be referred to as the set of *interior* aligned positive basis directions. For brevity these sets may be referred to as either the exterior and interior sets or the exterior and interior directions.

Distance to a blocking constraint. Let $c_i(x) = a_i^\mathsf{T} x - b_i$ be a single linear inequality constraint and let $x_{\bar{p}}$ be the (orthogonal) projection of $x^{(k)}$ onto the line $c_i(x) = 0$ so that

$$x_{\bar{p}} = x^{(k)} - d_i \frac{a_i}{\|a_i\|}$$

for some $d_i \in \mathbb{R}$. By a Taylor series expansion

$$c_i(x_{\bar{p}}) = c_i(x^{(k)}) - d_i \frac{a_i^\mathsf{T} a_i}{\|a_i\|} \tag{4.4}$$

and since $c_i(x_{\bar{p}}) = 0$, re-arranging equation (4.4) gives

$$d_i = \frac{a_i^\mathsf{T} x^{(k)} - b_i}{\|a_i\|}.$$

Since a_i points into the feasible region of constraint c_i, if $d_i \geqslant 0$ then $x^{(k)}$ is feasible otherwise $x^{(k)}$ violates constraint c_i. The distance from $x^{(k)}$ to the line $c_i(x) = 0$ is given by $|d_i|$. Figure 4.3 graphically illustrates a simple two dimensional case.

4.2 Feasible line search

Since a constraint may block the progress of the line search it is necessary to modify line search Condition L1 for the constrained problem. If all the line search trial points

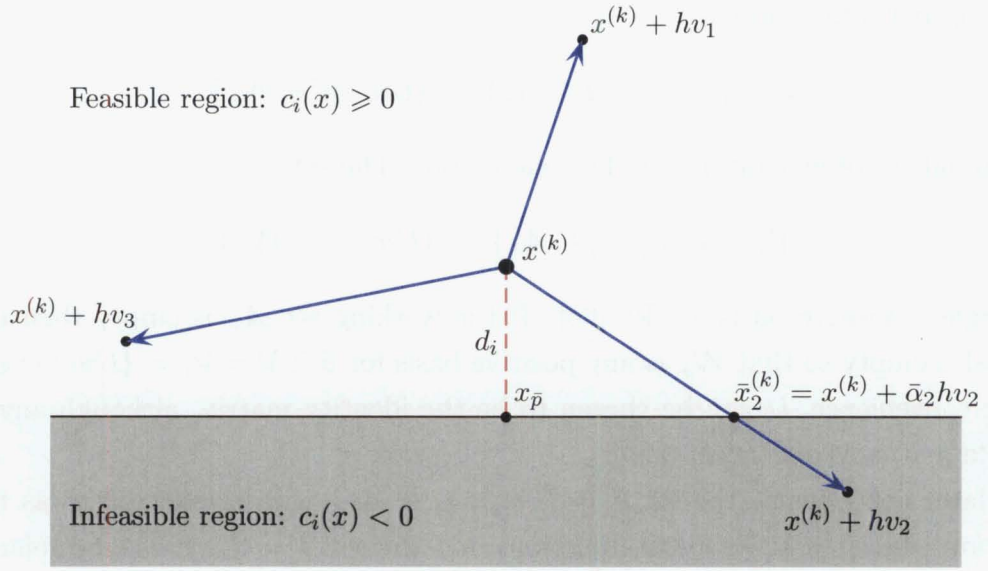

Figure 4.3: Orthogonal projection and an infeasible frame point.

are feasible then the revised line search condition L1′ is the same as Condition L1. However if a trial step α_j leads to the infeasibility of $x^{(k)} + \alpha_j h^{(k)} s^{(k)}$ then α_j is set to the largest feasible value so that

$$\alpha_j = \max\{\alpha \geqslant 0 : x^{(k)} + \alpha h^{(k)} s^{(k)} \in \Omega\}.$$

The maximum feasible step is discussed in more detail later in this section. At each iteration $\alpha^{(k)}$ is chosen to be the argument that minimises f over all (feasible) line search trial points. Hence $\alpha^{(k)} = \arg\min\{f_{j-1}, f_j\}$. If $f_{j-1} = f_j$ then $\alpha^{(k)}$ is set to the maximum feasible value α_j. As in Condition L1, $x^{(k)} + \alpha^{(k)} h^{(k)} s^{(k)}$ is returned as the outcome of the line search. Due to the possible presence of blocking constraints condition L1′ no longer ensures that a line local minimum lies in the α-interval $[\alpha_{j-2}, \alpha_j]$. However $\alpha^{(k)}$ is still the argument that minimises the objective function over all of the feasible line search trial points at which f is actually evaluated. If it is feasible, line search Condition L1′ will return the same point found by line search Condition L1. Otherwise Condition L1′ will return either the maximum feasible step ($\alpha^{(k)} = \alpha_j$) or the step immediately before the maximum feasible step ($\alpha^{(k)} = \alpha_{j-1}$).

Maximum feasible step. As well as illustrating the orthogonal projection of the current iterate $x^{(k)}$ onto a constraint Figure 4.3 also shows that frame point $x^{(k)} + hv_2$ is infeasible. If the maximum feasible step from $x^{(k)}$ in the direction v_2 is denoted by

$\bar{x}_2^{(k)} = x^{(k)} + \bar{\alpha}_2 h v_2$ then by a Taylor series expansion

$$c_i(\bar{x}_2^{(k)}) = c_i(x^{(k)}) + \bar{\alpha}_2 h a_i^{\mathsf{T}} v_2 = 0$$

so that

$$\bar{\alpha}_2 = \frac{b_i - a_i^{\mathsf{T}} x^{(k)}}{h a_i^{\mathsf{T}} v_2}.$$

More generally, if $a_i^{\mathsf{T}} v_j \neq 0$ then

$$\bar{\alpha}_{ij} = \frac{b_i - a_i^{\mathsf{T}} x^{(k)}}{h a_i^{\mathsf{T}} v_j}$$

is the step-length from $x^{(k)}$ in the direction v_j so that $\bar{x}_{ij}^{(k)} = x^{(k)} + \bar{\alpha}_{ij} h v_j$ satisfies $c_i(x) = 0$. If $\bar{\alpha}_{ij}$ is positive then v_j points towards $c_i(x) = 0$. If $\bar{\alpha}_{ij} = 0$ then $x^{(k)}$ satisfies $c_i(x) = 0$ and if $\bar{\alpha}_{ij}$ is negative then v_j points away from $c_i(x) = 0$. If $a_i^{\mathsf{T}} v_j = 0$ then movement in the directions $\pm v_j$ is parallel to $c_i(x) = 0$.

4.3 Simple descent frame-based line search algorithms

Each iteration of the simple descent frame-based line search algorithm for linearly constrained optimisation presented in this section follows the same approach as the unconstrained algorithm of Template U1 except that aligned positive bases and feasible line searches are used. It is assumed that only the function values at feasible points are obtainable. The adequate decrease condition of equation (3.1) is modified for the linearly constrained algorithm whenever a constraint blocks the progress of the line search. In such a situation the active constraint working set $\mathcal{A}_+^{(k)}$ is updated to include such a blocking constraint and feasible line searches are performed along the directions $s_{\mathfrak{E}}^{(k)}$ and $s_{\mathfrak{I}}^{(k)}$ from the exterior and interior sets such that $x^{(k)} + h^{(k)} s_{\mathfrak{E}}^{(k)}$ and $x^{(k)} + h^{(k)} s_{\mathfrak{I}}^{(k)}$ are feasible and

$$f(x^{(k)} + h^{(k)} s_{\mathfrak{E}}^{(k)}) \leqslant f(x^{(k)} + h^{(k)} v)$$

for all $v \in \mathfrak{E}^{(k)}$ such that $x^{(k)} + h^{(k)} v \in \Omega$ and

$$f(x^{(k)} + h^{(k)} s_{\mathfrak{I}}^{(k)}) \leqslant f(x^{(k)} + h^{(k)} v)$$

for all $v \in \mathfrak{I}^{(k)}$ such that $x^{(k)} + h^{(k)} v \in \Omega$. If the decrease in the objective function due to a feasible line search along $s_{\mathfrak{E}}^{(k)}$ is denoted by $\Delta_{\mathfrak{E}}^{(k)}$ and the decrease in the objective function due to a feasible line search along $s_{\mathfrak{I}}^{(k)}$ is denoted by $\Delta_{\mathfrak{I}}^{(k)}$ then the

search direction $s^{(k)}$ can be any feasible direction such that a feasible line search along $s^{(k)}$ produces descent at least as large as $\max\{\Delta_{\mathfrak{E}}^{(k)}, \Delta_{\mathfrak{J}}^{(k)}\}$. This is prevents the failure of the algorithm whenever progress in a good *local* descent direction, as defined by equation (3.1), is repeatedly blocked by a constraint. Just as in the unconstrained case, the step $s^{(k)}$ is known as the adequate descent step. For later convenience, if the active constraint working set $\mathcal{A}_+^{(k)}$ is empty, so that the set of exterior directions $\mathfrak{E}^{(k)}$ is also empty, then $\Delta_{\mathfrak{E}}^{(k)}$ is set to negative infinity.

Template C1: Linearly constrained simple descent algorithm

Step 1: Set $k = 0$ and choose the initial point $x^{(0)}$.

Step 2: Choose $h^{(k)}$, calculate an aligned positive basis $\mathcal{V}_+^{(k)}$ and choose $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$.

Step 3: Determine a search direction $s^{(k)}$.

Step 4: Set $\Delta^{(k)} = f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)})$.

Step 5: If $\Delta^{(k)} > 0$ then:

Perform a feasible line search and let $z^{(k)}$ be the point selected by that search.

Otherwise:

Set $z^{(k)} = x^{(k)}$.

Step 6: Calculate f at a finite number of feasible points and choose $x^{(k+1)}$ to be the lowest of these points and $z^{(k)}$.

Step 7: If stopping conditions are satisfied then stop, otherwise increment k and go to Step 2.

Each $\mathcal{S}_+^{(k)}$ is a positive spanning set for the feasible region at $x^{(k)}$ since it contains the aligned positive basis $\mathcal{V}_+^{(k)}$ as a subset. Each $\mathcal{S}_+^{(k)}$ must contain only finitely many points in order to ensure that Step 3 is a finite process. Apart from restrictions on length the choice of vectors which are added to $\mathcal{V}_+^{(k)}$ to form $\mathcal{S}_+^{(k)}$ is arbitrary however only feasible points may be considered when determining the search direction in Step 3.

The choice of the finite set of feasible points in Step 6 is also arbitrary but any such point cannot be accepted as the next iterate unless it is better than the point $z^{(k)}$ chosen in Step 5. If no extra function values are calculated in Step 6 then a pair of identical iterates is generated whenever a minimal frame occurs.

Convergence of the algorithm

The convergence of algorithm Template C1 is determined by examining the asymptotic properties of arbitrary subsequences of iterates when the stopping conditions are never invoked. The following revised assumptions are required to prove convergence of the linearly constrained algorithms.

Assumptions (revised).

A1′. $f : \mathbb{R}^n \to \mathbb{R}$ is a $\mathcal{C}^1$ objective function.

A2′. The sequence of all iterates $\{x^{(k)}\}$ is contained in a compact subset of the feasible region Ω.

A3′. $\{h^{(k)}\}$ is a sequence of frame size parameters such that $h^{(k)} \to 0$ as $k \to \infty$.

A4′. $\{\mathcal{V}_+^{(k)}\}$ is a sequence of aligned positive bases so that subsets of $\{\mathcal{V}_+^{(k)}\}$ generate all possible cones of feasible directions at (and near) $x^{(k)}$.

A5′. There exist positive constants ξ and Ξ such that $\xi \leqslant \|v\| \leqslant \Xi$ for each $v \in \mathcal{V}_+^{(k)}$ and for each $k \in \mathbb{N}$.

A6′. There exists a positive constant τ_θ such that for each $k \in \mathbb{N}$ and any (non-zero) feasible direction $g \in \mathbb{R}^n$

$$\max_{v \in \mathcal{V}_+^{(k)}} \left\{ \frac{-v^\mathsf{T} g}{\|v\| \cdot \|g\|} \right\} \geqslant \tau_\theta.$$

A7′. $0 < \|s^{(k)}\| \leqslant \Xi$ for each $k \in \mathbb{N}$.

A8′. The line search complies with Condition L1′.

Note that Assumptions A1′, A3′, A5′ and A7′ are unchanged from the unconstrained assumptions on page 26. The only difference in Assumption A6′ is that now g must be a feasible direction.

As the vectors $\{u_q : i_q \in \mathcal{A}_+\} \subseteq \mathcal{V}_+$ are constructed according to the procedure in part (a) of the constructing aligned positive bases section (on page 44) it is necessary to show that Assumption A5′ still holds for the constrained case. By equation (4.2) (on page 44), $u_q^\mathsf{T} a_{i_q} = 1$ and so by the Cauchy-Schwarz [70] inequality

$$\|u_q\| \geqslant \frac{1}{\|a_{i_q}\|}.$$

Since there are only finitely many constraints there are only finitely many constraint normals so there exists one with maximum norm. Hence there exists a positive constant ξ such that $\|u_q\| \geqslant \xi$ for each $q \in \{1, \ldots, |\mathcal{A}_+|\}$. However, as even the following simple two dimensional example shows, it is possible to construct the matrix U so that $\|u_q\|$ is arbitrarily large for some $q \in \{1, \ldots, |\mathcal{A}_+|\}$.

Example 4.1. Consider the two dimensional case with only one active constraint whose non-zero constraint normal is $a_1 = (\gamma_1, \gamma_2)^\mathsf{T}$. The matrix

$$U = \frac{1}{\gamma_1\gamma_4 - \gamma_2\gamma_3} \begin{pmatrix} \gamma_4 & -\gamma_2 \\ -\gamma_3 & \gamma_1 \end{pmatrix}$$

is chosen arbitrarily apart from the restriction that $a_1 = (\gamma_1, \gamma_2)^\mathsf{T}$ and $(\gamma_3, \gamma_4)^\mathsf{T}$ are linearly independent so that $\gamma_1\gamma_4 - \gamma_2\gamma_3 \neq 0$. By construction $U^\mathsf{T} a_1 = (1, 0)^\mathsf{T}$ however

$$\|u_1\|^2 = \frac{\gamma_3^2 + \gamma_4^2}{(\gamma_1\gamma_4 - \gamma_2\gamma_3)^2}$$

can be made arbitrarily large even if the value of $\gamma_1\gamma_4 - \gamma_2\gamma_3$ remains fixed.

The solution presented in [66] is that the upper bound Ξ is imposed retrospectively. If the norm of u_q exceeds Ξ for any $q \in \{1, \ldots, |\mathcal{A}_+|\}$ then u_q is appropriately scaled. This does not alter the important properties of u_q: namely, u_q still points into the feasible region of constraint c_{i_q} and u_q remains orthogonal to all the other constraint normals in the current working set. Although this is one area that would benefit from further research. For example, it is possible that a finite precision implementation of this approach may fail under the following circumstances: if, before scaling, the norm of u_q (for some $q \in \{1, \ldots, |\mathcal{A}_+|\}$) is extremely large then after scaling $u_q^\mathsf{T} a_{i_q}$ may be evaluated as *zero* in finite precision arithmetic. Hence u_q may appear orthogonal to a_{i_q} instead of pointing into the feasible region of constraint c_{i_q} as intended by the construction. Further investigation into alternate constructions for aligned positive bases, although important, is beyond the scope of this thesis.

Convergence proof. It is shown in Theorem 4.1 that cluster points of the sequence of iterates generated by the simple descent frame-based line search algorithm for linearly constrained optimisation presented in Template C1 are KKT points of $\mathcal{C}^1$ objective functions under mild conditions.

Theorem 4.1. *Given Assumptions A1′–A8′ (on page 49) hold, cluster points of the sequence of iterates generated by algorithm Template C1 are* KKT *points of the objective function.*

Proof. Let $\{y^{(k)}\} \subseteq \{x^{(k)}\}$ be any subsequence of iterates generated by algorithm Template C1 such that $y^{(k)} \to y^{(\infty)}$ as $k \to \infty$. Since $y^{(k)} \to y^{(\infty)}$ as $k \to \infty$ there exists $K \in \mathbb{N}$ such that $\mathcal{A}_+^{(k)} \supseteq \mathcal{A}(y^{(\infty)})$ whenever $k \geqslant K$. Suppose that $y^{(\infty)}$ is not a KKT point. By Assumptions A4′–A6′ there exists $\Phi > 0$ and some $v^{(k)} \in \mathcal{V}_+^{(k)}$ which is feasible at $y^{(\infty)}$ and satisfies $v^{(k)\mathsf{T}} \nabla f(y^{(\infty)}) < -\Phi$, whenever $k \geqslant K$. By the $\mathcal{C}^1$ continuity of f, for any $\beta \in (0,1)$ there exists $L \geqslant K$ such that $v^{(k)\mathsf{T}} \nabla f(y^{(k)}) < -\beta\Phi$ whenever $k \geqslant L$. Since $y^{(k)} \to y^{(\infty)}$ as $k \to \infty$ and the nearest constraint to $y^{(\infty)}$ not already in $\mathcal{A}(y^{(\infty)})$ is some distance from $y^{(\infty)}$ there exists $\delta > 0$ such that a feasible line search can take a step of length at least $\min\{\delta, \tau_{\text{LS}_1}\}$ whenever $k \geqslant L$. Using the same approach as Lemma 3.4, the uniform bounds on the directional derivative and the guaranteed minimum step length ensure that $\max\{\Delta_{\mathfrak{C}}^{(k)}, \Delta_{\mathfrak{I}}^{(k)}\}$ is uniformly bounded away from zero whenever $k \geqslant L$. Hence $f(y^{(k+1)}) - f(y^{(k)}) \not\to 0$ as $k \to \infty$, a contradiction of the $\mathcal{C}^1$ continuity of f, so that $y^{(\infty)}$ is a KKT point of the objective function f. □

As the objective function decreases monotonically distinct cluster points have the same function value. Furthermore, since the cluster points are the limit points of subsequences in a compact subset of the feasible region the cluster points are also feasible. In the next section a sufficient descent algorithm that allows opportunistic movement to a new feasible iterate whenever the objective function is reduced by a sufficient amount is presented.

4.4 Sufficient descent frame-based line search algorithms

At each iteration of the sufficient descent frame-based line search algorithm for linearly constrained optimisation candidate search directions from a finite set $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$, which contains the aligned positive basis $\mathcal{V}_+^{(k)}$ as a subset, are examined. Just as in algorithm Template C1 it is assumed that only the function values at feasible points are obtainable. A feasible search direction $s^{(k)} \in \mathcal{S}_+^{(k)}$ is accepted without the need to complete a frame about the current iterate whenever the sufficient descent condition of equation (3.8) on page 34 is satisfied and $x^{(k)} + h^{(k)}s^{(k)}$ is feasible. If $\Delta^{(k)} > \epsilon^{(k)}$ then a feasible line search is conducted along $x^{(k)} + \alpha h^{(k)} s^{(k)}$ ($\alpha > 0$) and the point $z^{(k)} = x^{(k)} + \alpha^{(k)} h^{(k)} s^{(k)}$ is returned as the line search's outcome. Otherwise $\Delta^{(k)} \leqslant \epsilon^{(k)}$ for every $s \in \mathcal{S}_+^{(k)}$ such that $x^{(k)} + h^{(k)}s$ is feasible and the algorithm proceeds exactly like the simple descent algorithm of Template C1: either a search direction is chosen

so that a line search produces descent at least as large as that due to a line search along the best exterior and interior directions, or $\Delta^{(k)} \leqslant 0$ and the frame is minimal: the current iterate is relabelled $z^{(k)}$ and the frame size is reduced. The algorithm may take any feasible point not higher than $z^{(k)}$ as the next iterate $x^{(k+1)}$. This completes an iteration. The algorithm halts when the stopping conditions are satisfied.

Template C2: Linearly constrained sufficient descent algorithm

Step 1: Set $k = 0$ and choose the initial point $x^{(0)}$.

Step 2: Choose $h_{\min}^{(k)} > 0$ and $\epsilon_{\min}^{(k)} > 0$.

Step 3: Choose $h^{(k)} \geqslant h_{\min}^{(k)}$, $\epsilon^{(k)} \geqslant \epsilon_{\min}^{(k)}$, calculate an aligned positive basis $\mathcal{V}_+^{(k)}$ and choose $\mathcal{S}_+^{(k)} \supseteq \mathcal{V}_+^{(k)}$.

Step 4: For each $s^{(k)} \in \mathcal{S}_+^{(k)}$ satisfying $x^{(k)} + h^{(k)}s^{(k)} \in \Omega$:
Calculate $\Delta^{(k)} = \max\{f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)})\}$ until $\Delta^{(k)} > \epsilon^{(k)}$ or every $s^{(k)} \in \mathcal{S}_+^{(k)}$ satisfying $x^{(k)} + h^{(k)}s^{(k)} \in \Omega$ has been used.

Step 5: If $\Delta^{(k)} > \epsilon^{(k)}$ then (a sufficient descent step exists):
Perform a feasible line search and let $z^{(k)}$ be the point selected by that search.
Otherwise (the frame is $\epsilon^{(k)}$-quasi-minimal):
(a) Determine a search direction $s^{(k)}$.
(b) Set $\Delta^{(k)} = f(x^{(k)}) - f(x^{(k)} + h^{(k)}s^{(k)})$.
(c) If $\Delta^{(k)} > 0$ then:
Perform a feasible line search and let $z^{(k)}$ be the point selected by that search.
Otherwise (the frame is minimal):
Set $z^{(k)} = x^{(k)}$.

Step 6: Calculate f at a finite number of feasible points and choose $x^{(k+1)}$ to be the lowest of these points and $z^{(k)}$.

Step 7: If stopping conditions are satisfied then stop.

Step 8: If $\Delta^{(k)} > \epsilon^{(k)}$ then:
Set $h_{\min}^{(k+1)} = h_{\min}^{(k)}$ and $\epsilon_{\min}^{(k+1)} = \epsilon_{\min}^{(k)}$. Increment k and go to Step 3.
Otherwise:
Increment k and go to Step 2.

Each $\mathcal{S}_+^{(k)}$ is a positive spanning set for the feasible region at $x^{(k)}$ since it contains the aligned positive basis $\mathcal{V}_+^{(k)}$ as a subset. Each $\mathcal{S}_+^{(k)}$ must contain only finitely many points in order to ensure that Step 4 is a finite process. The choice of vectors which are added to $\mathcal{V}_+^{(k)}$ to form $\mathcal{S}_+^{(k)}$ is arbitrary apart from restrictions length but only feasible points may be considered in Step 4.

There are two possibilities at Step 5: either a feasible sufficient descent step is known in advance, or the frame is quasi-minimal and a line search is attempted along any search direction that produces descent at least as large as that produced by a line search along the best exterior and interior directions. That is, a line search is either performed along *any* feasible direction that produces a sufficient decrease step, or a quasi-minimal frame exists and the algorithm attempts to find a search direction using the same procedure as used in algorithm Template C1.

The choice of the finite set of feasible points in Step 6 is also arbitrary but any such point cannot be accepted as the next iterate unless it is better than the point $z^{(k)}$ chosen in Step 5. If no extra function values are calculated in Step 6 then a pair of identical iterates is generated whenever a minimal frame occurs.

Convergence of the algorithm

Just like algorithm Template U2 the algorithm described in Template C2 can be viewed as two nested loops. The outer loop selects positive values for the minimum frame size $h_{\min}^{(k)}$ and the minimum sufficient descent parameter $\epsilon_{\min}^{(k)}$. The inner loop locates an $\epsilon^{(k)}$-quasi-minimal frame centre while the values of $h_{\min}^{(k)}$ and $\epsilon_{\min}^{(k)}$ remain fixed. A new arbitrary frame size $h^{(k)}$ and sufficient descent parameter $\epsilon^{(k)}$ can only be chosen once a frame has been completed about the current iterate. Until that time the frame size and sufficient descent parameter cannot be reduced below the lower bounds $h_{\max}^{(k)}$ and $\epsilon_{\min}^{(k)}$. Hence, just as in algorithm Template U2, the inner loop of algorithm Template C2 is a finite process.

Theorems 4.2 and 4.3 are the linearly constrained equivalents of Theorems 3.2 and 3.3 for the unconstrained algorithms. Theorem 4.2 states that the sequence of quasi-minimal frames is infinite.

Theorem 4.2. *Given Assumptions A1′–A8′ (on page 49) hold, algorithm Template C2 generates an infinite sequence of quasi-minimal frame centres.*

Proof outline. The proof is very similar to that of Theorem 3.2 and has already been outlined at the beginning of this section: the inner loop of algorithm Template C2 is a finite process; hence failure to find a sufficient descent step, and thus the creation

of a quasi-minimal frame, must occur infinitely often so that algorithm Template C2 generates an infinite sequence of quasi-minimal frame centres. □

Theorem 4.3 states that cluster points of the sequence of quasi-minimal frame centres generated by algorithm Template C2 are KKT points of $\mathcal{C}^1$ objective functions whenever the sufficient descent parameter $\epsilon^{(k)} = o(h^{(k)})$.

Theorem 4.3. *Given:*

(a) *Assumptions A1′–A8′ (on page 49) hold,*

(b) $\epsilon^{(k)} = o(h^{(k)})$,

cluster points of the sequence of quasi-minimal frame centres generated by algorithm Template C2 are KKT *points of the objective function.*

Proof outline. As algorithm Template C2 proceeds exactly like algorithm Template C1 whenever sufficient descent is not obtained (that is, whenever a quasi-minimal frame is created about the current iterate) the proof for Theorem 4.1 holds so that cluster points of the sequence of quasi-minimal frame centres generated by algorithm Template C2 are KKT points of the objective function. □

Chapter 5
Grid local minimisers and stopping conditions

From the sufficient conditions presented in Chapter 1, and provided sufficient derivative information is available, x^* is a solution of the unconstrained optimisation problem

$$\min_{x \in \mathbb{R}^n} f(x) \tag{5.1}$$

whenever $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. For all but very simple problems the direct calculation of such a solution is not possible and so iterative methods must be used. Since an algorithm's asymptotic convergence properties can only be established in the absence of stopping conditions it is not uncommon for these to receive all but superficial treatment when an algorithm is presented. However appropriate stopping conditions may be vital to ensure that good theoretical performance is maintained in practice. If derivative information is unavailable it can be difficult to determine whether the sequences $\{x^{(k)}\}$ and $\{f(x^{(k)})\}$ have converged to a minimiser x^* and minimum $f(x^*)$. It can be unwise to base such deductions on the progress made during a set of iterations as this progress can be exceedingly erratic [3, p. 8].

Typically there are two tests which determine the stopping criteria for derivative-free optimisation algorithms. The first is that the change in function values from one iteration to the next is sufficiently small so that

$$|f(x^{(k)}) - f(x^{(k-1)})| \leqslant \tau_f$$

for some non-negative constant τ_f. The second is that successive estimates of the minimiser are sufficiently close so that

$$\|x^{(k)} - x^{(k-1)}\| \leqslant \tau_x$$

for some non-negative constant τ_x. Note that these criteria may be satisfied far from a minimiser resulting in the termination of an algorithm before an accurate approximation to the solution is attained [3, p. 9].

Several recent papers discuss the use of pattern search [29, 74] and grid-based methods [22, 23] to solve the unconstrained optimisation problem (5.1) whenever derivative information is unavailable. As discussed in Chapter 2 the basic idea behind such methods is to find an analogue of a stationary point when the function is restricted to

the nodes of a grid (or mesh) for a sequence of progressively finer grids. It has been shown in [22, 74] that under mild conditions any limit point of such a sequence is a stationary point of the function. Both of these papers ([22, 74]) include the case where convergence is proven for the sequence of grid points for which no adjacent grid point has lower function value. Such points are called *grid local minimisers* in [22, 23] and *unsuccessful* iterates in [29, 74].

It is common practice for pattern search, grid- and frame-based optimisation algorithms to terminate when the equivalent of a grid local minimiser has been found and the mesh, grid or frame size parameter is sufficiently small. These criteria will be referred to as the *standard* stopping conditions for pattern search, grid- and frame-based methods. In this chapter some of the consequences of relying solely on these standard stopping conditions for pattern search and grid-based algorithms are investigated and some extra conditions which ensures their effectiveness in practice are presented. The results obtained in this chapter are generalisable for the more flexible frame-based algorithms discussed in Chapters 3 and 4, however in this chapter only the simpler structure provided by grid-based methods is considered. This simplifies the presentation and prevents the effects of a sequence of different positive bases from clouding the issues discussed.

Although the number of grid local minimisers and their location does not affect the theoretical properties of grid-based methods it may have a significant effect in practice. If a sequence of grid local minimisers is located far from the minimiser the grid size may be reduced rapidly and prematurely. Under such conditions an algorithm may take steps which are tiny requiring many iterations for any significant progress towards the minimiser [22]. For a given function and grid it is in general difficult to determine the number and position of grid local minimisers without evaluating the function at each of the grid points. For a strictly convex quadratic function one may intuitively think that only the grid points closest to the minimiser will be grid local minimisers. However this is false, even in two dimensions for general grids, but true for some grids including those based on conjugate directions.

From a theoretical point of view the choice of grid and thus the underlying positive basis is irrelevant. However for practical finite-precision arithmetic implementations the grid-size parameter never tends to zero and so limiting processes cannot be relied upon. As maximal positive bases sample the objective function at the largest number of grid points surrounding the current iterate they must, in general, produce the best local models of the objective function. Figure 5.1 shows part of a regular orthogonal grid in two dimensions and three positive bases. Positive basis (a) is the minimal positive basis

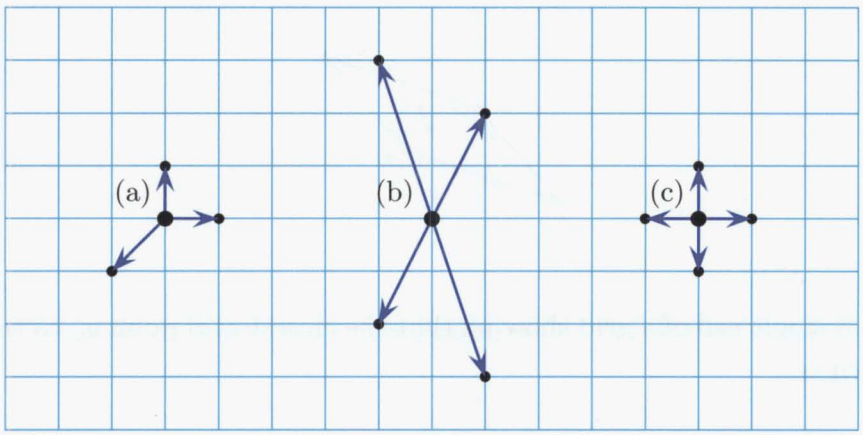

Figure 5.1: Selected positive bases on a regular orthogonal grid.

used in Example 2.1 (on page 12), positive basis (b) is a maximal positive basis chosen at random and positive basis (c) is the maximal positive basis used in Example 2.2 (on page 12). Positive basis (c) allows the value of the objective function to be investigated at the largest number of grid points (for a positive basis); all of which are as close to the current iterate as the grid allows. Positive basis (c) is therefore a good positive basis to use in the sense that it is unlikely that any other positive basis could, in general, produce a better local model of the objective function about the current iterate. The material in the remainder of this chapter deals exclusively with grid-based methods for unconstrained optimisation using maximal positive bases of type (c): for a given basis $\mathcal{V} = \{v_i\}_{i=1}^n$ only grids associated with the maximal positive basis $\mathcal{V}_+ = \{\pm v_i\}_{i=1}^n$ will be considered. The justification is that if this choice of positive basis is shown to have the potential to produce poor results when used in conjunction with the standard stopping conditions for pattern search and grid-based methods then it is unlikely that any other positive basis will, in general, do any better.

As defined in Chapter 2 a grid point $\breve{x}$ is a grid local minimiser or GLM for the objective function f on the grid $\mathcal{G}(x_0, h, \mathcal{V})$ with respect to the positive basis $\mathcal{V}_+$ if

$$f(\breve{x} + hv) \geqslant f(\breve{x}), \quad \forall v \in \mathcal{V}_+.$$

The following additional definitions are used throughout the remainder of this chapter.

Definition 5.1 (Grid directions). The vectors v_i are referred to as the grid directions.

Definition 5.2 (Grid line). A grid line is a line through one of the grid points in one of the grid directions.

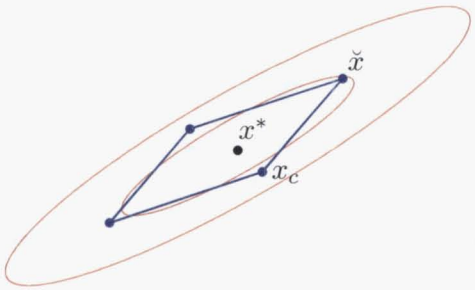

Figure 5.2: A single cell of a grid showing that the closest grid point x_c to the minimiser x^* is not a GLM.

Definition 5.3 (Adjacent grid points). The vectors $\pm hv_i$ are the steps between adjacent grid points.

Depending on the shape of the grid the closest grid point (using the standard Euclidean metric) to the minimiser x^* may not be a GLM. A two dimensional example is illustrated in Figure 5.2 which shows the contours of a strictly convex quadratic function and a single cell of a grid. The closest grid point x_c to the minimiser x^* is not a GLM.

A new metric is now introduced to measure the distance between two points relative to the grid $\mathcal{G}(x_0, h, \mathcal{V})$. Since the set $\mathcal{V} = \{v_i\}_{i=1}^n$ is a basis for $\mathbb{R}^n$ any point can be uniquely expressed as a linear combination of the grid directions.

Definition 5.4 (Grid-norm). The grid-norm of a point $x = h\sum_{i=1}^n \zeta_i v_i$, where $\zeta_i \in \mathbb{R}$ is defined as

$$\|x\|_{\mathcal{G}} = h\sum_{i=1}^{n} |\zeta_i| \cdot \|v_i\|$$

where $\|\cdot\|$ represents the standard Euclidean norm.

Definition 5.5 (Grid-distance). The grid-distance between any two points x and y is $\|x - y\|_{\mathcal{G}}$.

The grid-distance measures the shortest distance between two points when travel is restricted to the grid directions. Some authors refer to this as the *taxi-cab* metric. Note that there may not be a unique closest grid point to a given point $x \in \mathbb{R}^n$ due to the symmetry of the grid about x.

As the diagonals of each cell of a given grid are the same length using the grid-distance metric the diameter of a cell of a grid can be defined in a natural way.

Definition 5.6 (Cell diameter). The cell diameter of the grid $\mathcal{G}(x_0, h, \mathcal{V})$ is

$$\operatorname{diam}(\mathcal{G}) = h\sum_{i=1}^{n} \|v_i\|.$$

5.1 Number of grid local minimisers

Figure 5.3 shows the contours of Rosenbrock's function [68] and the number and position of GLMs with a square grid centred on the origin with grid size parameter $h = 1/3$. Figure 5.4 shows a strictly convex quadratic function and an orthogonal grid where the only GLM is the minimiser of the function. One may intuitively think that only the grid points closest to the minimiser will be GLMs for such nicely behaved functions. Figure 5.5 shows that this is false even in two dimensions. Figure 5.6 shows an example of a two dimensional non-orthogonal grid where the only GLM is the minimiser. All illustrations using a strictly convex quadratic function in two dimensions are based on the *example quadratic* function $q(x, y) = x^2 + 25y^2$. Occasionally the function used will be the example quadratic rotated about the origin (as in Figures 5.5 and 5.6).

The number and position of GLMs is in general difficult to calculate. However for a strictly convex quadratic function in two dimensions and a particular class of grid it is possible to give an (attainable) upper bound for the maximum distance from a GLM to the minimiser and a formula for the total number of GLMs. The particular class of grids considered throughout the remainder of this chapter are called *diagonally aligned* grids and are defined as follows.

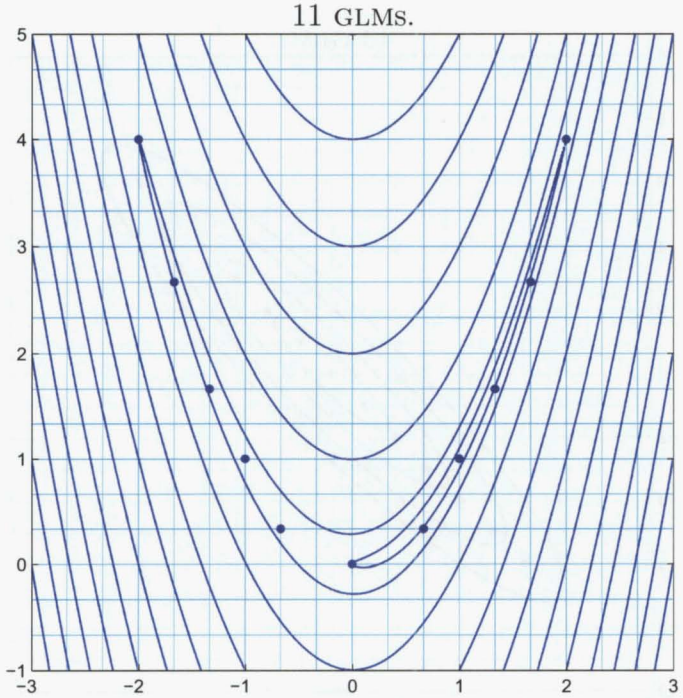

Figure 5.3: Rosenbrock's function and orthogonal grid showing the number and location of GLMs.

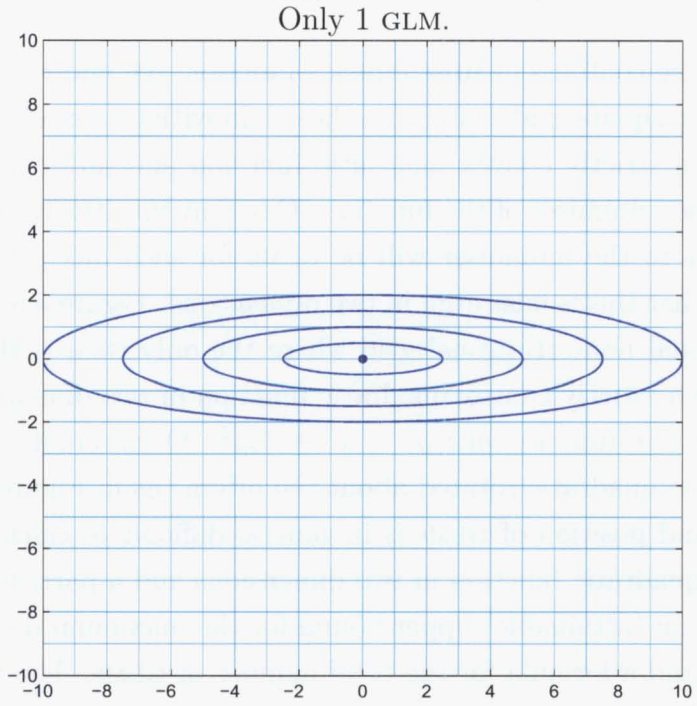

Figure 5.4: Orthogonal grid with a strictly convex quadratic.

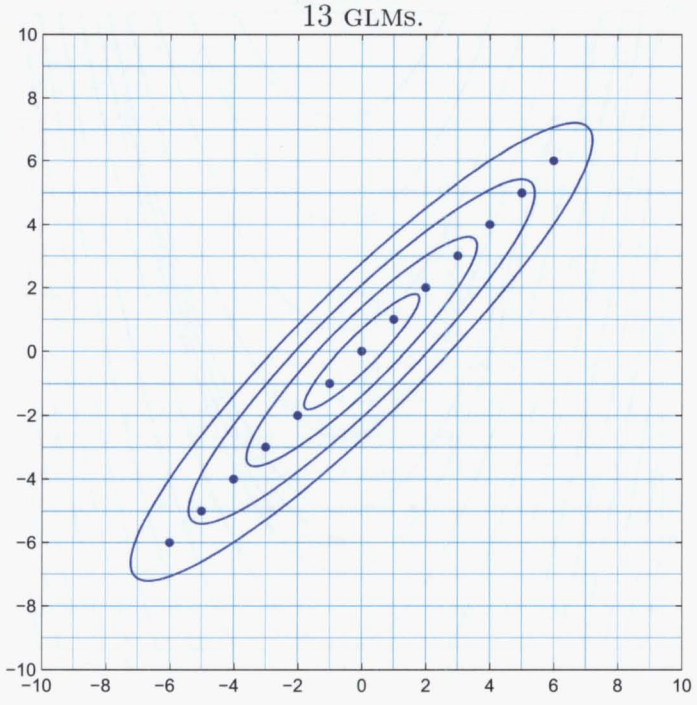

Figure 5.5: Orthogonal grid with a (rotated) strictly convex quadratic.

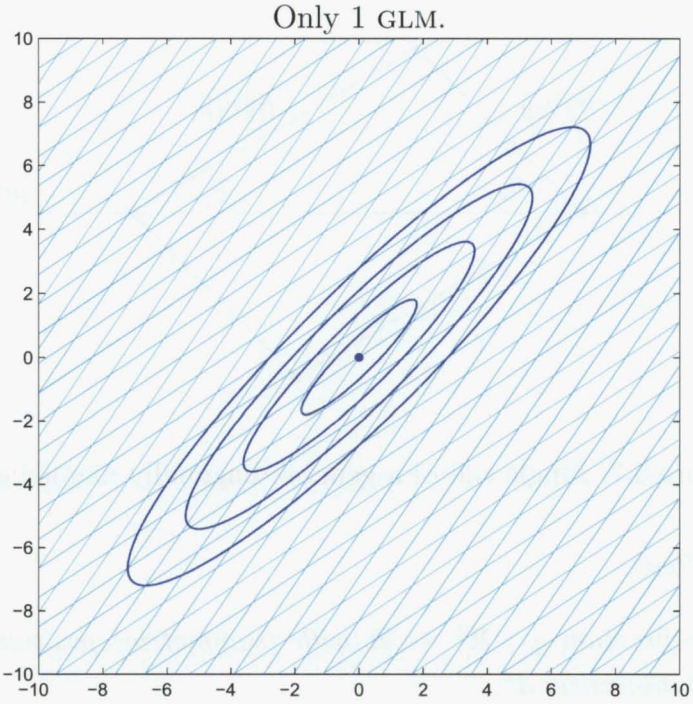

Figure 5.6: Non-orthogonal grid with a (rotated) strictly convex quadratic.

Definition 5.7 (Diagonally aligned grid). A diagonally aligned grid for a given strictly convex quadratic function in two dimensions is one in which:

(a) The minimiser of the quadratic function is a grid point.

(b) The principal axis of the quadratic function is parallel to one of the diagonals of the grid's cells (parallelograms in two dimensions).

Figure 5.7 illustrates a single cell of a general diagonally aligned grid. For such a grid let θ_1 and θ_2 be the angles between the grid directions and the principal axis of the quadratic function. Assume $\theta_1 \leqslant \theta_2$ so that $\theta_1 \in (0, \pi/2)$ and $\theta_2 \in [\theta_1, \pi - \theta_1)$ (if $\theta_1 = 0$ the grid collapses to a line). Clearly $\|v_1\| \sin\theta_1 = \|v_2\| \sin\theta_2$ and the distance between grid points along the principal axis is $\breve{d} = h(\|v_1\| \cos\theta_1 + \|v_2\| \cos\theta_2)$. The only way that no component of a grid direction is towards the minimiser is if the grid line is orthogonal to the principal axis. As the grid directions must remain linearly independent at least one of the grid directions cannot be orthogonal to the principal axis. Hence for the remainder of this discussion assume $\theta_1 \leqslant \theta_2$ with $\theta_1 \in (0, \pi/2)$ and $\theta_2 \in [\theta_1, \pi/2]$. The following definition is used in the proof of Theorem 5.1 below.

Definition 5.8 (Critical grid point). A grid point is critical when one of its adjacent grid points has the same function value.

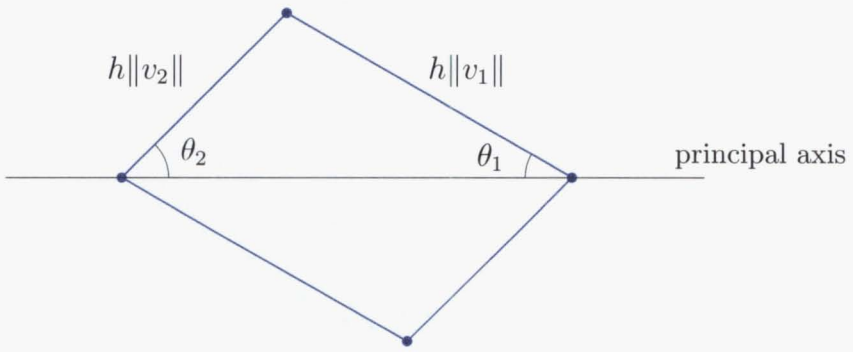

Figure 5.7: Single cell of a general diagonally aligned grid.

Theorem 5.1. *Given:*

(a) *any quadratic function* $q : \mathbb{R}^2 \to \mathbb{R}$ *with (symmetric) positive definite Hessian matrix* B *and minimiser* x^*,

(b) *a diagonally aligned grid with parameters* h, θ_1, θ_2, v_1 *and* v_2,

(c) *a* GLM $\breve{x}$ *on the principal axis,*

the maximum distance from the minimiser x^* *to a* GLM $\breve{x}$ *on the principal axis is*

$$\max \|x^* - \breve{x}\| = \frac{h}{2}\|v_i\| \sin\theta_i \left(\kappa \tan\theta_i + \cot\theta_i\right)$$

where

$$i = \begin{cases} 1 & \text{if } \kappa \geqslant \cot\theta_1 \cot\theta_2 \\ 2 & \text{otherwise} \end{cases}$$

and $\kappa = \mu_{max}/\mu_{min}$ *is the condition number of the matrix* B *whose largest eigenvalue* $\mu_{max} = 1/b^2$ *and smallest eigenvalue* $\mu_{min} = 1/a^2$.

Proof. The contours of q are ellipses centred on x^*. With an appropriate coordinate system these contours have equation $(x/a)^2 + (y/b)^2 = (t/a)^2$ with $x \in [0, t]$ when restricted to the right half-plane (symmetry takes care of the other half-plane). Suppose $\breve{x} = (t, 0)$ is a grid point on the principal axis and that C is the contour line of q which passes through $\breve{x}$. If $\breve{x}$ is a GLM then none of the adjacent grid points have lower function value (lie inside C). The critical case which determines the maximum value for t, so that $\breve{x}$ is a GLM, occurs when none of the grid points adjacent to $\breve{x}$ lie inside C but at least one of the adjacent grid points lies on C. For each $i \in \{1, 2\}$ let t_i

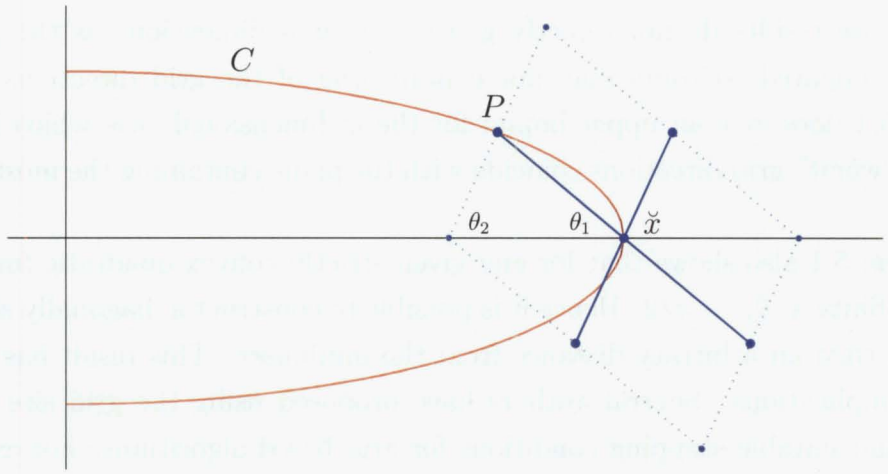

Figure 5.8: Maximum distance grid local minimiser.

be a critical value of t so that $\breve{x}$ is a critical grid point. Let $P^{(i)} = (P_x^{(i)}, P_y^{(i)})$ be the corresponding critical grid point adjacent to $\breve{x}$ so that $P_x^{(i)} = t_i - h\|v_i\| \cos\theta_i$ and $P_y^{(i)} = h\|v_i\| \sin\theta_i$. Since $P^{(i)} = (P_x^{(i)}, P_y^{(i)})$ is on the elliptical contour C it follows that $a^2(P_y^{(i)})^2 = b^2(t_i^2 - (P_x^{(i)})^2)$ and so $t_i = h\|v_i\| \sin\theta_i \left(\kappa \tan\theta_i + \cot\theta_i\right)/2$ (the case for $i = 1$ is illustrated in Figure 5.8). As t increases from zero the maximum distance between a GLM and the minimiser is determined by whichever grid point adjacent to $\breve{x}$ becomes critical first. Hence

$$\max \|x^* - \breve{x}\| = \min_{i \in \{1,2\}} \{t_i\}.$$

By direct calculation $t_1 \leqslant t_2$ whenever $\kappa \geqslant \cot\theta_1 \cot\theta_2$. □

Perhaps even more important from a practical point of view is that there is also a minimum bound on the distance of the furthest GLM from the minimiser. Let $t_{\max}$ be the maximum distance along the principal axis a GLM can be from the minimiser. The following corollary follows immediately from Theorem 5.1.

Corollary 5.1. *Given the conditions of Theorem 5.1 hold, the distance from the minimiser to the furthest* GLM *on the principal axis is at least*

$$\max\{0, t_{max} - \breve{d}\}$$

where $\breve{d} = h(\|v_1\| \cos\theta_1 + \|v_2\| \cos\theta_2)$ *is the distance between grid points along the principal axis.*

Proof. If $t_{\max} < \breve{d}$ then the only GLM on the principal axis is the minimiser. However if $t_{\max} \geqslant \breve{d}$ there must be a GLM on the principal axis whose distance from the minimiser is in the interval $(t_{\max} - \breve{d}, t_{\max}]$. □

The above results do not directly generalise for n dimensions as the plane with the most elongated contours may not contain *any* of the grid directions. However Theorem 5.1 does give an upper bound for the n-dimensional case which is attained when the "worst" grid directions coincide with the plane containing the most elongated contours.

Theorem 5.1 also shows that for any given strictly convex quadratic function $t_{\max}$ tends to infinity as $\theta_1 \to \pi/2$. Hence it is possible to construct a diagonally aligned grid that has a GLM an arbitrary distance from the minimiser. This result has important practical implications. Several authors have proposed using the grid size parameter to determine suitable stopping conditions for grid-based algorithms. For example the author of [74, p. 9] states:

> ... in the absence of any explicit higher-order information about the function to be minimized, it makes sense to terminate a generalized pattern search algorithm when $\Delta^{(k)}$ [the grid size parameter] is less than some reasonably small tolerance. In fact, this is a common stopping condition for algorithms of this sort ...

and from [49, p. 931]:

> The stopping criteria that seems most natural to us is to halt the algorithm once $\Delta^{(k)}$ [the grid size parameter] falls below some prescribed tolerance Δ_*.

Theorem 5.1 and Corollary 5.1 show that unless some extra precautions are taken using the grid size parameter alone does not guarantee that the minimiser (of even a strictly convex quadratic function in two dimensions) is within *any* given distance of the final iterate. As observed in [77, p. 196]:

> For any non-derivative method, the issue of termination is problematical as well as highly sensitive to problem scaling. Since gradient information is unavailable, it is provably impossible to verify closeness to optimality simply by sampling f at a finite number of points.

The following example using a diagonally aligned grid illustrates the danger of terminating an algorithm at a GLM using only the grid size parameter as the stopping criterion.

Example 5.1. Let $V = [v_1, v_2]$ be the matrix whose columns are the grid directions v_1 and v_2. Many practical grid-based algorithms require $\det(V) \geqslant \tau_V$ for some small

positive constant τ_V. Suppose the grid directions are $v_1 = [\tau_V, 1]^\mathsf{T}$ and $v_2 = [0, 1]^\mathsf{T}$ so that $\|v_1\| = \csc\theta_1$, $\|v_2\| = 1$ and $\cot\theta_1 = \tau_V$. Since $\det(V) = \tau_V$ the grid is acceptable and since $\cot\theta_1 \cot\theta_2 = 0$ Theorem 5.1 gives

$$\begin{aligned} t_{\max} &= \frac{h}{2}\|v_1\| \sin\theta_1(\kappa \tan\theta_1 + \cot\theta_1) \\ &= \frac{h}{2\tau_V}\left(\kappa + \tau_V^2\right) \\ &\approx \frac{h\kappa}{2\tau_V} \quad \text{(for small } \tau_V\text{)}. \end{aligned}$$

If $\kappa = 100$ and $\tau_V = 10^{-6}$ then $t_{\max} \approx 5 \times 10^7 h$. So that even for a strictly convex quadratic function with condition number 100 an algorithm that terminates at a GLM with stopping conditions based solely on the gird size parameter h could terminate when the distance from the minimiser to the current iterate is over seven orders of magnitude larger than h. Note that $\tau_V = 10^{-6}$ is not an overly demanding choice and could be much smaller in practice. The convergent variant of the Nelder-Mead algorithm described in [67] for example uses 10^{-18} as the lower bound on the linear independence of the simplex directions. Clearly this is a bad grid for this problem. However such information is rarely available in advance (compare the grids used in Figures 5.5 and 5.6 for example). Furthermore whilst the transformation $x' = V^{-1}x$ orthogonalises the grid the condition number of the transformed problem increases so much that $t'_{\max} \approx 10^{14}h$.

Forcing the grid to be orthogonal does not produce good results in practice (consider the performance of the method of alternating variables (which no one seems to want to take responsibility for) or the method of Hooke and Jeeves [44]). The above results go some way to explaining why. With a diagonally aligned regular orthogonal grid $\tan\theta_1 = 1 = \tan\theta_2$ and so $t_{\max} \approx \kappa h/2$. Hence an algorithm could terminate at a GLM when the distance from the minimiser to the final iterate is many orders of magnitude larger than h. Theorem 5.1 also provides the framework for the following corollary.

Corollary 5.2. *Given:*

(a) *any strictly convex quadratic function* $q : \mathbb{R}^2 \to \mathbb{R}$,

(b) *a diagonally aligned grid with parameters* h, θ_1, θ_2, v_1 *and* v_2,

the total number of GLMs *is*

$$1 + 2\left\lfloor \frac{t_{max}}{\breve{d}} \right\rfloor \tag{5.2}$$

where $\breve{d} = h(\|v_1\| \cos\theta_1 + \|v_2\| \cos\theta_2)$ *is the distance between grid points along the principal axis.*

Proof. Theorem 5.1 shows that if $\breve{x} = (t, 0)$ is a grid point on the principal axis with $|t| \leqslant t_{\max}$ then $\breve{x}$ is a GLM as all adjacent grid points lie either outside or on the contour line through $\breve{x}$. However if $|t| > t_{\max}$ then at least one grid point adjacent to $\breve{x}$ will lie inside the contour line though $\breve{x}$. Hence for a grid point $\breve{x}$ on the principal axis $\breve{x}$ is a GLM if and only if $|t| \leqslant t_{\max}$. The minimiser is a grid point by the definition of diagonally aligned grids and so by symmetry the total number of GLMs is $1 + 2\lfloor t_{\max}/\breve{d} \rfloor$. □

Corollary 5.2 establishes that:

(a) It is possible to construct a diagonally aligned grid with an arbitrary number of GLMs for any strictly convex quadratic function even in two dimensions.

(b) The number of GLMs is independent of the grid size parameter.

The results of Corollary 5.2 do not directly generalise to higher dimensions. In n dimensions GLMs may lie in some bounded region of an $n-1$ dimensional hyperplane. In this situation the parameter $t_{\max}$ corresponds to the size of the bounded region. The following example illustrates this situation in three dimensions.

Example 5.2. Consider the function $q(x, y, z) = x^2 + y^2 + z^2 + 2M(x + y + z)^2$ for some large positive integer M and a unit cubic grid centred on the origin and aligned with the coordinate axes. Let $\breve{x} = (x_1, y_1, z_1)$ be a GLM on the plane $P : x + y + z = 0$. If $\bar{x}$ is a grid point adjacent to $\breve{x}$ then $\bar{x} = (x_1 \pm 1, y_1, z_1)$ or $\bar{x} = (x_1, y_1 \pm 1, z_1)$ or $\bar{x} = (x_1, y_1, z_1 \pm 1)$. Suppose $\bar{x} = (x_1 \pm 1, y_1, z_1)$ then $|x_1| \leqslant M$ implies $q(\breve{x}) \leqslant q(\bar{x})$. By symmetry the same bound also applies to y_1 and z_1. Hence grid points that lie in this bounded region of the plane P are GLMs. For this example there are approximately $3M^2$ GLMs on the plane P the furthest of which is a distance $M\sqrt{2}$ from the minimiser. Note that there may also be other GLMs that do not lie on the plane P.

Clearly there can be *very* many more GLMs in higher dimensions. To make matters even worse as the dimensionality increases a greater proportion of these GLMs will be close to the maximum distance from the minimiser, a very bad situation in practice [62].

5.2 Conjugate grids

The examples of the previous section clearly illustrate the potential problems of terminating a grid-based algorithm at a GLM whenever the grid size parameter reaches some pre-determined cut-off value. In this section it is shown that if the grid directions are based on conjugate directions then the standard stopping conditions can be used with confidence. Grids based on conjugate directions are defined as follows.

Definition 5.9 (Conjugate grid). For a given strictly convex quadratic function with positive definite Hessian matrix B, a B-conjugate grid is one in which the grid directions are conjugate with respect to the Hessian matrix B.

For a given quadratic function there are infinitely many B-conjugate grids. When the quadratic function is not in question the shorter term *conjugate grid* may be used. For strictly convex quadratic functions grids based on conjugate directions guarantee that only the closest grid points to the minimiser will be GLMs. Multiple GLMs can only exist if the grid is symmetric about the minimiser and in this case all such GLMs:

(a) Belong to the same cell of the grid.

(b) Have the same function value.

(c) Are the same (grid) distance from the minimiser.

These claims are proved in the following theorem and its corollary.

Theorem 5.2. *Given:*

(a) *any quadratic function* $q : \mathbb{R}^n \to \mathbb{R}$ *with (symmetric) positive definite Hessian matrix* B *and minimiser* x^*,

(b) *a set of* B*-conjugate vectors* $\mathcal{V} = \{v_i\}_{i=1}^n$ *which define the grid* $\mathcal{G}(x_0, h, \mathcal{V})$,

then

$$\breve{x} \text{ is a GLM} \iff \|x^* - \breve{x}\|_{\mathcal{G}} \leqslant \frac{1}{2}\text{diam}(\mathcal{G}).$$

Proof. Write $x^* = \breve{x} + h\sum_{i=1}^n \zeta_i v_i$, where $\zeta_i \in \mathbb{R}$ and $q(x) = (x^* - x)^{\mathsf{T}}B(x^* - x)/2 + c$ for some constant $c \in \mathbb{R}$. The grid point $\breve{x}$ is a GLM if and only if $q(\breve{x}) \leqslant q(x)$ for all x adjacent to $\breve{x}$. The grid point x is adjacent to $\breve{x}$ if and only if $x = \breve{x} \pm hv_k$ for some $k \in \{1, \ldots, n\}$. Hence by conjugacy

$$q(\breve{x}) \leqslant q(x) \iff \zeta_k^2 v_k^{\mathsf{T}} B v_k \leqslant (\zeta_k \pm 1)^2 v_k^{\mathsf{T}} B v_k \iff |\zeta_k| \leqslant 1/2. \tag{5.3}$$

Since equation (5.3) holds for every x adjacent to $\breve{x}$, $|\zeta_i| \leqslant 1/2$ for all $i \in \{1, \ldots, n\}$ and so by Definitions (5.4) and (5.6) (on page 58)

$$\|x^* - \breve{x}\|_{\mathcal{G}} = h \sum_{i=1}^{n} |\zeta_i| \cdot \|v_i\| \leqslant \frac{1}{2} \mathrm{diam}(\mathcal{G}). \qquad \square$$

The next corollary follows immediately from Theorem 5.2.

Corollary 5.3. *Given:*

(a) *the conditions of Theorem 5.2 hold,*

(b) $\breve{x}$ *is a* GLM,

then

$$q(\breve{x}) \leqslant q(x), \quad \forall x \in \mathcal{G}$$

and

$$\|x^* - \breve{x}\|_{\mathcal{G}} \leqslant \|x^* - x\|_{\mathcal{G}}, \quad \forall x \in \mathcal{G}.$$

Proof. Write $x = \breve{x} + h \sum_{i=1}^{n} \eta_i v_i$, where $\eta_i \in \mathbb{Z}$ and $x^* = \breve{x} + h \sum_{i=1}^{n} \zeta_i v_i$, where $\zeta_i \in \mathbb{R}$. From the proof of Theorem 5.2, $|\zeta_i| \leqslant 1/2$ for each $i \in \{1, \ldots, n\}$ so that $|\zeta_i| > |\zeta_i - \eta_i|$ has no solutions for $\eta_i \in \mathbb{Z}$. Hence $|\zeta_i| \leqslant |\zeta_i - \eta_i|$ for all $i \in \{1, \ldots, n\}$ so that

$$q(\breve{x}) - q(x) = \frac{h^2}{2} \sum_{i=1}^{n} \zeta_i^2 v_i^{\mathsf{T}} B v_i - \frac{h^2}{2} \sum_{i=1}^{n} (\zeta_i - \eta_i)^2 v_i^{\mathsf{T}} B v_i \leqslant 0$$

and

$$\|x^* - \breve{x}\|_{\mathcal{G}} = h \sum_{i=1}^{n} |\zeta_i| \cdot \|v_i\| \leqslant h \sum_{i=1}^{n} |\zeta_i - \eta_i| \cdot \|v_i\| = \|x^* - x\|_{\mathcal{G}}. \qquad \square$$

For a strictly convex quadratic function and an associated conjugate grid Theorem 5.2 and Corollary 5.3 show that every GLM is within half a cell diameter of the minimiser. Furthermore if *any* GLM is found then no other grid point can be closer to the minimiser (using the grid distance metric) or have lower function value.

Smallest angle between pairs of conjugate directions

Conjugate directions for grid-based methods have nice properties relating the proximity of GLMs to the minimiser of strictly convex quadratic functions. However as shown in previous chapters avoiding a degenerate sequence of positive bases is vital to ensure the convergence of grid- and frame-based methods. For a given strictly convex

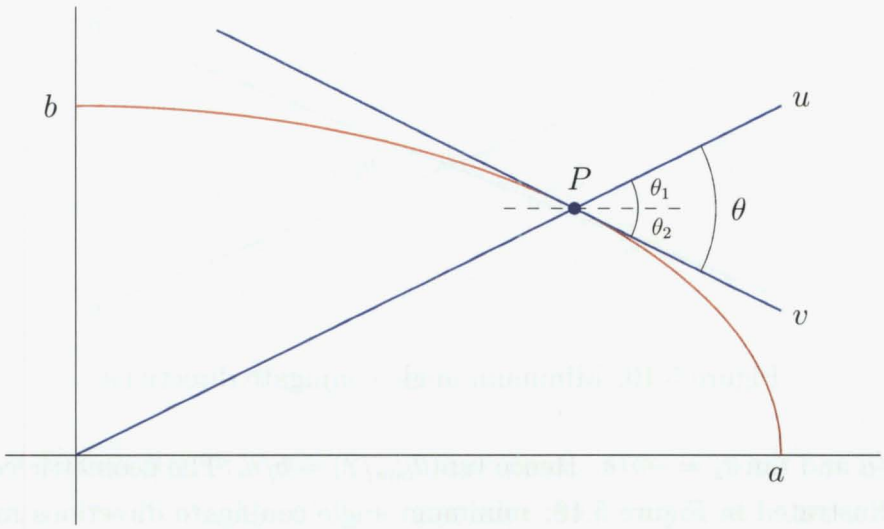

Figure 5.9: Angle between conjugate directions.

quadratic function the use grids based on conjugate directions also ensures that degenerate sequences of positive bases are avoided. Although there are infinitely many sets of conjugate directions for a given strictly convex quadratic function there exists a smallest (non-zero) angle between pairs of conjugate directions.

Theorem 5.3. *Given any quadratic function $q : \mathbb{R}^n \to \mathbb{R}$ with (symmetric) positive definite Hessian matrix B, the smallest angle between any pair of B-conjugate directions is*

$$\theta_{min} = 2\tan^{-1}(\kappa^{-\frac{1}{2}})$$

where $\kappa = \mu_{max}/\mu_{min}$ is the condition number of B whose largest and smallest eigenvalues are μ_{max} and μ_{min}.

Proof. Firstly the two dimensional case. Consider the ellipse $(x/a)^2 + (y/b)^2 = 1$ restricted to the first quadrant as shown in Figure 5.9 (symmetry takes care of the other quadrants). If $a = b$ the ellipse is a circle and the angle between any pair of conjugate directions is $\pi/2$. Suppose $a > b > 0$ and consider the line u through the origin and a point $P = (P_x, P_y)$ on the ellipse. If $P_x = 0$ or $P_x = a$ then the corresponding conjugate direction is orthogonal to u. If $P_x \in (0, a)$ then u has slope $\tan\theta_1 = P_y/P_x$ and the corresponding conjugate direction v has slope $\tan\theta_2 = \mathrm{d}y/\mathrm{d}x|_{P_x}$. The angle $\theta = \theta_1 - \theta_2$ between u and v is given by

$$\tan\theta = \frac{\tan\theta_1 - \tan\theta_2}{1 + \tan\theta_1 \tan\theta_2}.$$

Minimising θ for $x \in (0, a)$ is equivalent to minimising $\tan\theta_1 - \tan\theta_2$ since $\tan\theta_1 \tan\theta_2 = -b^2/a^2$ is constant. The minimum occurs when $x = a/\sqrt{2}$ so that

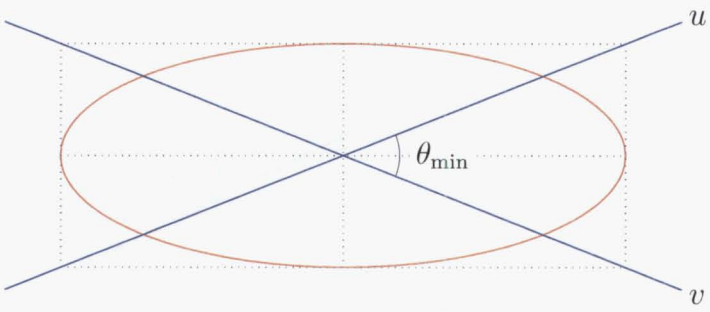

Figure 5.10: Minimum angle conjugate directions.

$\tan\theta_1 = b/a$ and $\tan\theta_2 = -b/a$. Hence $\tan(\theta_{\min}/2) = b/a$. The geometric consequence of this is illustrated in Figure 5.10: minimum angle conjugate directions intersect the vertices of the bounding rectangle (with sides parallel to the major and minor axes) of the elliptical contours.

The ellipse $(x/a)^2 + (y/b)^2 = 1$ can be considered as a single contour of the two dimensional quadratic $q(x) = x^{\mathsf{T}}Bx/2$ for the (symmetric) positive definite matrix

$$B = \begin{pmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{pmatrix}$$

with eigenvalues $\mu_{\min} = 1/a^2$ and $\mu_{\max} = 1/b^2$. Hence

$$\tan\left(\frac{\theta_{\min}}{2}\right) = \sqrt{\frac{\mu_{\min}}{\mu_{\max}}}. \tag{5.4}$$

Since any two conjugate directions define a plane and the minimum angle between pairs of conjugate directions depends on the elongation of the elliptical contours, not on the orientation of the ellipses, this result generalises nicely for higher dimensions. The set of all possible smallest angles between pairs of conjugate directions is minimised when the (elliptical) contours are at their most elongated and this occurs along the major and minor axes so that equation (5.4) holds for n dimensions. □

Smallest angle conjugate directions also have nice properties if finite differences are used to approximate gradient or second order information. If the (conjugate) grid directions are orthogonal then an ill-conditioned function is more sensitive to perturbations in certain directions: similar sized steps may not produce similar sized changes in function values. However for pairs of smallest angle conjugate directions the function is equally sensitive along each of the conjugate directions. Note that the non-orthogonal grid shown in Figure 5.6 (on page 61) uses the smallest angle conjugate directions for the example quadratic function.

Chapter 6

Effects of limited precision on selected quasi-Newton methods

The calculation of derivative information is either difficult or impossible for many practical optimisation problems; but this is not always the case. It would be advantageous if the methods described in this thesis were able to use derivative information effectively if it became available. This chapter investigate the performance of a conjugate directions factorisation implementation of BFGS and DFP quasi-Newton methods. It is shown that such an implementation outperforms the other methods considered (including Cholesky factorisation) when approximate second-order information is only available to limited precision. This suggests that grid- and frame-based algorithms which use conjugate directions in their positive bases are also able to calculate quasi-Newton steps that are effective in practice whenever derivative information is available.

Quasi-Newton algorithms are used to solve the local optimisation problem

$$\min_{x \in \mathbb{R}^n} f(x)$$

iteratively where $f : \mathbb{R}^n \to \mathbb{R}$ and gradient information is available. A solution x^* is attained when $\nabla f(x^*) = 0$ but in practice the usual requirement is that $\|\nabla f(x^*)\| \leqslant \tau_g$ for some (typically small) non-negative constant τ_g.

The development of quasi-Newton or variable metric algorithms, as they were originally called, is attributed to Davidon [25] in 1959 and became popularised as the "DFP" method by Fletcher and Powell [32] in 1963. This method was found to work well in practice when used in conjunction with accurate line searches. However the DFP method is less effective when used with the low accuracy line searches which have become popular since the 1970s.

Work by Broyden [5, 6, 7], Fletcher [30], Goldfarb [38], Shanno [71], and also Greenstadt [41] led to the development of the "BFGS" method. In practice BFGS outperforms DFP when used with low accuracy line searches. There are many alternative quasi-Newton update formulae, for example "SR1" the symmetric rank one update and the (infinitely large) Broyden family of updates of which BFGS and DFP are both members.

Throughout this chapter the convention of writing $\nabla f(x^{(k)})$ as $g^{(k)}$ is used. At iteration k of a quasi-Newton method a search direction $p^{(k)}$ is found by solving the

system of equations

$$B^{(k)}p^{(k)} = -g^{(k)} \tag{6.1}$$

where $B^{(k)}$ approximates, in some sense, the Hessian matrix $\nabla^2 f(x^{(k)})$. A line search is then performed along $x^{(k)} + \alpha p^{(k)}$ where $\alpha \in \mathbb{R}$ to find a new iterate $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$ for some $\alpha^{(k)}$ that satisfies some line search criteria. Information at this new point is used to generate a new approximate Hessian matrix $B^{(k+1)}$. If $B^{(k)}$ is positive definite then $p^{(k)\mathsf{T}} g^{(k)} < 0$ so that $p^{(k)}$ is a descent direction for f. In this situation the line search is replaced by a ray search ($\alpha > 0$).

The use of Cholesky factorisations of the approximate Hessian matrices $B^{(k)}$ was introduced in [35] and is now in widespread use (it is coded as `VA13A` in the Harwell subroutine library [43] for example). Proponents of this implementation claim it avoids the computational instability of using the inverses of the approximate Hessian matrices and allows the efficient calculation of the search direction in $\mathcal{O}(n^2)$ operations by using forward and back substitution. The standard Cholesky factorisation implementation of the BFGS method uses the *modified* Cholesky factorisation $B^{(k)} = L^{(k)} D^{(k)} L^{(k)\mathsf{T}}$ where $L^{(k)}$ is unit lower triangular and $D^{(k)}$ is diagonal. The modified implementation allows the easy detection (and subsequent correction) of loss of positive definiteness of the approximate Hessian matrices (due to rounding errors in finite precision arithmetic) with little extra computational effort. As the theory of Cholesky factorisations is well established (see for example [4, 36, 69]) it is not discussed further here.

This chapter examines the performance of a selection of BFGS and DFP implementations on a suite of ill-conditioned test problems across a range of dimensions and line search criteria as the precision of second-order information varies from 16 to two digits. The results support those in [39], specifically, that there is no numerical evidence to support the claim that a Cholesky factor implementation of the BFGS formula offers any improvement in performance, as is popularly believed, over more straightforward implementations when second-order information is available to full precision. Furthermore these results are extended to show that a factorisation strategy has clear advantages when second-order information is only available to limited precision. However a Cholesky factorisation is not necessarily the best one to use. Evidence is also obtained to support the claim that the numerical instability of non-factored implementations of quasi-Newton methods reported by some authors is due to early implementations of the DFP formula with low accuracy line searches.

6.1 BFGS and DFP formulae

The BFGS and DFP update formulae can be written as

$$\text{(BFGS)} \qquad B^{(k+1)} = \left[B + \frac{\gamma\gamma^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} - \frac{Bss^{\mathsf{T}}B}{s^{\mathsf{T}}Bs} \right]^{(k)} \tag{6.2}$$

and

$$\text{(DFP)} \qquad B^{(k+1)} = \left[B + \left(1 + \frac{s^{\mathsf{T}}Bs}{s^{\mathsf{T}}\gamma} \right) \frac{\gamma\gamma^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} - \left(\frac{\gamma s^{\mathsf{T}}B + Bs\gamma^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} \right) \right]^{(k)} \tag{6.3}$$

where $s^{(k)} = x^{(k+1)} - x^{(k)}$ and $\gamma^{(k)} = g^{(k+1)} - g^{(k)}$. Note that the iteration superscript k applies to each of the variables inside the square brackets. If the inverse of $B^{(k)}$ is denoted by $H^{(k)}$ then application of the Sherman-Morrison-Woodbury [56, 72, 76] formula gives

$$\text{(BFGS)} \qquad H^{(k+1)} = \left[H + \left(1 + \frac{\gamma^{\mathsf{T}}H\gamma}{s^{\mathsf{T}}\gamma} \right) \frac{ss^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} - \left(\frac{s\gamma^{\mathsf{T}}H + H\gamma s^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} \right) \right]^{(k)} \tag{6.4}$$

and

$$\text{(DFP)} \qquad H^{(k+1)} = \left[H + \frac{ss^{\mathsf{T}}}{s^{\mathsf{T}}\gamma} - \frac{H\gamma\gamma^{\mathsf{T}}H}{\gamma^{\mathsf{T}}H\gamma} \right]^{(k)}. \tag{6.5}$$

Equations (6.4) and (6.5) allow the direct calculation of the search direction without the need to solve the system of equations (6.1). The implementations discussed in this chapter fall into three categories:

(a) Direct updates of the approximate Hessian matrices $B^{(k)}$.

(b) Direct updates of the inverses of the approximate Hessian matrices $H^{(k)}$.

(c) Factorisations: either Cholesky factorisations of the approximate Hessian matrices or conjugate factorisations of their inverses.

Conjugate factorisation

The method of conjugate factorisation used in this chapter is based on [17] however the idea in not new (see for example [24, 59, 60]). A brief description follows but see [17] for more details.

The BFGS update formula (6.4) can be written in product form [4] as

$$H^{(k+1)} = \left[(I - p\mathfrak{q}^{\mathsf{T}})H(I - p\mathfrak{q}^{\mathsf{T}})^{\mathsf{T}}\right]^{(k)}$$

where

$$\mathfrak{q}^{(k)} = \left[\frac{\gamma}{p^{\mathsf{T}}\gamma} \pm \frac{g}{\sqrt{-p^{\mathsf{T}}g\, p^{\mathsf{T}}\gamma/\alpha}}\right]^{(k)}.$$

If the inverse Hessian approximation matrices are factored so that $H^{(k)} = C^{(k)}C^{(k)\mathsf{T}}$ then the columns of $C^{(k)}$ are $B^{(k)}$-conjugate and the search direction $p^{(k)} = -C^{(k)}d^{(k)}$ where $d^{(k)} = C^{(k)\mathsf{T}}g^{(k)}$. The elements of $d^{(k)}$ are the directional derivatives of f at $x^{(k)}$ in the directions of the columns of $C^{(k)}$. The updated conjugate factors can be written as

$$C^{(k+1)} = \left[C - \frac{p\mathfrak{z}^{\mathsf{T}}}{p^{\mathsf{T}}\gamma} \mp \frac{pd^{\mathsf{T}}}{\sqrt{-p^{\mathsf{T}}g\, p^{\mathsf{T}}\gamma\,/\alpha}}\right]^{(k)} \tag{6.6}$$

where $\mathfrak{z}^{(k)} = C^{(k)\mathsf{T}}\gamma^{(k)}$ is the difference in the directional derivatives at $x^{(k+1)}$ and $x^{(k)}$. Then $d^{(k+1)} = C^{(k+1)\mathsf{T}}g^{(k+1)}$ can be written as

$$d^{(k+1)} = \bar{d}^{(k)} - \frac{p^{(k)\mathsf{T}}g^{(k+1)}\,\mathfrak{z}^{(k)}}{p^{(k)\mathsf{T}}\gamma^{(k)}} \mp \frac{p^{(k)\mathsf{T}}g^{(k+1)}\,d^{(k)}}{\sqrt{-p^{(k)\mathsf{T}}g^{(k)}\,p^{(k)\mathsf{T}}\gamma^{(k)}\,/\alpha^{(k)}}} \tag{6.7}$$

where $\bar{d}^{(k)} = C^{(k)\mathsf{T}}g^{(k+1)}$. Equations (6.6) and (6.7) can be written in terms of the new variables d and $\mathfrak{z}$ so that

$$C^{(k+1)} = \left[C - \frac{p\mathfrak{z}^{\mathsf{T}}}{d^{\mathsf{T}}\mathfrak{z}} \mp \frac{pd^{\mathsf{T}}}{\sqrt{-d^{\mathsf{T}}d\, d^{\mathsf{T}}\mathfrak{z}\,/\alpha}}\right]^{(k)}$$

and

$$d^{(k+1)} = \left[\bar{d} - \frac{\bar{d}^{\mathsf{T}}d\,\mathfrak{z}}{d^{\mathsf{T}}\mathfrak{z}} \mp \frac{\bar{d}^{\mathsf{T}}d\,d}{\sqrt{-d^{\mathsf{T}}d\, d^{\mathsf{T}}\mathfrak{z}\,/\alpha}}\right]^{(k)}.$$

There are two obvious implementations, one for each of the $+/-$ signs in equation (6.6). After limited numerical trials [9] both implementations were found to perform very similarly. The implementation presented in this chapter uses the $+$ sign from equation (6.6).

Implementations

There are many ways to implement the BFGS and DFP formulae presented in equations (6.2)–(6.5). The four BFGS and DFP implementations considered here were selected as a representative sample from the 12 BFGS and 10 DFP implementations considered in [9]. As all of the numerical results were obtained using MATLAB [52], many of MATLAB's built-in functions were used for convenience. Text in `typewriter` font is used to emphasize MATLAB code. The initial Hessian approximation (or its inverse) was set to the identity matrix for each of the implementations. The four BFGS implementations are:

Bupdate. Uses equation (6.2) to update the sequence of approximate Hessian matrices $B^{(k)}$. The search direction is calculated by using the MATLAB matrix inverse function via the equation

$$\mathtt{p}^{(\mathtt{k})} = -\mathtt{inv}(\mathtt{B}^{(\mathtt{k})}) * \mathtt{g}^{(\mathtt{k})}.$$

Note that direct inversion of the $B^{(k)}$ matrices is not recommended in practice due to the computational expense and its poor numerical stability. It is used here to provide a guideline for the worst performance that would be expected from this type of implementation. Interestingly however, limited numerical trials [9] showed that direct calculation of the inverse performed almost identically to more preferred implementations, using Gaussian elimination, for example.

Hupdate. Uses equation (6.4) to update the sequence of inverses of the approximate Hessian matrices $H^{(k)}$. The search direction is calculated directly via

$$\mathtt{p}^{(\mathtt{k})} = -\mathtt{H}^{(\mathtt{k})} * \mathtt{g}^{(\mathtt{k})}.$$

Cholesky. Uses a sequence of Cholesky factors $L^{(k)}$ which are updated (rather than recomputed from scratch) at each iteration. The particular implementation presented here uses MATLAB's Cholesky factor update command `cholupdate`. The search direction is calculated with forward and back substitution via

$$\mathtt{p}^{(\mathtt{k})} = -\mathtt{L}^{(\mathtt{k})\mathtt{T}} \backslash (\mathtt{L}^{(\mathtt{k})} \backslash \mathtt{g}^{(\mathtt{k})}).$$

Conjugate. Conjugate factorisation of the inverse approximate Hessian matrices using the plus sign from equation (6.6). The search direction is then obtained by direct calculation.

DFP implementations. Each of the DFP implementations is the DFP equivalent of the corresponding BFGS implementation.

The implementation BUPDATE requires $\mathcal{O}(n^3)$ operations at each iteration to update the second-order information and compute a new search direction whereas the remaining implementations require only $\mathcal{O}(n^2)$ operations. Additionally the (modified) Cholesky factorisation implementation allows the easy detection of loss of positive definiteness of the approximate Hessian matrices. The other implementations do not have this feature. However with a conjugate factorisation it is extremely unlikely that the inverse approximate Hessian matrices will lose positive definiteness. The worst that can happen is that they may become positive semi-definite. In fact Powell makes the comment in [60] that:

> We even find that, if we let Z [the conjugate factorisation matrix] be singular initially, then in practice the rounding errors of a sequence of updating calculations remove the singularity very successfully.

Thus if positive definiteness of the inverse approximate Hessian matrices is lost then it is extremely likely it will be restored at the next iteration—or the other way around—it is extremely unlikely that loss of positive definiteness will be maintained for any length of time if conjugate factors are used. Even the unlikely loss of positive definiteness can be detected in a computationally efficient way by using *triangular* conjugate factors. Such factors could be generated and updated at each iteration using a QR-factorisation for example [9].

6.2 Numerical results

Each of the four BFGS and DFP implementations described above were tested with two different line searches on the suite of 25 test functions listed in Tables 6.1 and 6.2 as the precision of the approximate Hessian information varied from 16 to two digits. The varying levels of precision were achieved by truncating the elements of the approximate Hessian matrices (possibly in factored form or their inverses) to the desired level. For example the elements of the matrix X are truncated to n digits with

$$\text{trunc}(X) = 10^{-d}\lfloor 10^d X \rfloor$$

where $d = n - \lceil \log_{10}(\max(|X|)) \rceil$ and $\lfloor\cdot\rfloor, \lceil\cdot\rceil$ represent the floor and ceiling functions.

Each of the higher dimensional tests listed in Table 6.2 was carried out in 8, 12, 20, 40 and 60 dimensions. The column labelled *Cond* in Table 6.1 represents the condition

Function	*Dim.*	*Initial point*	*Cond.*
Rosenbrock	2	$(-1.2, 1)$	2.5×10^{3}
Powell badly scaled	2	$(0, 1)$	2.1×10^{15}
Repeated Rosenbrock	4	$(-1.2, 1, -1.2, 1)$	2.5×10^{3}
Extended Rosenbrock	4	$(-1.2, 1, -1.2, 1)$	3.2×10^{3}
Powell singular	4	$(3, -1, 0, 1)$	∞

Table 6.1: Low dimension test functions.

Function	*Initial point*
Repeated Rosenbrock	$(-1.2, 1, -1.2, 1, \ldots)$
Extended Rosenbrock	$(-1.2, 1, -1.2, 1, \ldots)$
Powell singular	$(3, -1, 0, 1, \ldots)$
Hilbert quadratic	$(0, 0, 0, 0, \ldots)$

Table 6.2: Test functions for 8, 12, 20, 40 and 60 dimensions.

number of the Hessian matrices at the solution. Since the condition number is the ratio of the largest singular value to the smallest all of the Powell singular functions have infinite condition number. Increasing dimension does not alter the condition number of the repeated Rosenbrock functions and only slightly increases that of the extended Rosenbrock function which has condition number 3.6×10^3 for the 60-d case. The condition numbers of the Hilbert quadratics on the other hand are known to increase dramatically with increasing dimension. The 8-d Hilbert quadratic for example has condition number 1.5×10^{10} and this increases to 1.7×10^{16} in 12 dimensions. More details on the test functions can be found in [39, 40, 55].

A two-sided Wolfe line search was used so that at each iteration $\alpha^{(k)}$ was chosen so that $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$ satisfies

$$f(x^{(k+1)}) \leqslant f(x^{(k)}) + \rho\alpha^{(k)} p^{(k)\mathsf{T}} g^{(k)}$$

and

$$|\, p^{(k)\mathsf{T}} g^{(k+1)}| \leqslant \sigma |\, p^{(k)\mathsf{T}} g^{(k)}|$$

where the sufficient descent parameter $\rho = 10^{-4}$ and the gradient parameter σ was set to 10^{-3} and 0.9 for what are referred to in the remainder of this chapter as *strict* and *standard* line searches. The Wolfe line search was implemented using an iterative safeguarded parabolic interpolation scheme.

For each test problem the number of function evaluations, final function value and execution time (in seconds) was recorded. As algorithm execution time depends on the computing environment as well as the implementation the mean execution times

presented here are indicative only and not used in the ranking scheme. The implementations were ranked by the number of test problems that were successfully solved (out of a possible total of $25 \times 15 = 375$). A test problem was deemed to have been successfully solved if the termination criterion $\|\nabla f(x)\| \leqslant 10^{-6}$ was met. If necessary the algorithms were then subsorted by the mean number of function evaluations. Any ties were subsorted by the mean accuracy of the approximations to the minimum function values. The accuracy was measured using $\log_{10}(f - f^*)$ where f^* represents the minimum of the function and f is the final function value. Note that $f^* = 0$ for each of the test problems in Tables 6.1 and 6.2. Only data for the problems that were solved successfully were used in the sorting process. As it is the "raw" performance of each implementation that is being investigated the algorithms were terminated whenever they ran into difficulty rather than applying some sort of safeguarding or corrective procedure. The implementations were deemed unsuccessful and thus terminated if more than 10^5 function evaluations were required, a descent direction was not found, a step of zero length was calculated, the function values became unbounded (as a result of division by zero due to rounding errors in finite precision arithmetic), a factorisation failed (where appropriate) or the line search failed. The line search failed if the global limit of 10^5 function evaluations was reached, more than 10^3 parabolic interpolation iterations were required or a zero step was calculated. All of the implementations presented in this chapter were run in a MATLAB R13 [52] environment on a Sun-Fire-880 multi-user machine with four 750MHz processors and 8GB of RAM running Solaris 8.

In each of the following results tables the columns labelled *Succ*, *Fcnt*, *Accy* and *Time* represent the number of successfully solved test problems, the mean number of function evaluations, the mean accuracy of the solutions and the mean execution time in seconds.

Full precision second-order information

The performance of each implementation with full precision (16 digits) second-order information for the strict and standard line searches is presented in Tables 6.3 and 6.4. When successful all implementations produced similarly accurate approximations to the solutions of the test problems.

Strict line search. All of the implementations solved all 25 test problems. The mean number of function evaluations ranged from 313.6 for the BFGS implementation of HUPDATE through to 410.2 for the DFP implementation of BUPDATE. The mean number of function evaluations required by the BFGS implementations was 327 ± 13

Ranking	*Method*	*Succ*	*Fcnt*	*Accy*	*Time*
(BFGS) 1	HUPDATE	25	313.6	-13.9	0.4
2	CONJUGATE	25	316.0	-14.1	0.4
3	CHOLESKY	25	318.0	-13.9	0.4
4	BUPDATE	25	339.4	-14.2	0.5
(DFP) 1	CONJUGATE	25	388.0	-14.0	0.4
2	HUPDATE	25	388.7	-14.0	0.5
3	CHOLESKY	25	395.7	-13.9	0.5
4	BUPDATE	25	410.2	-14.0	0.6

Table 6.3: Strict line search and 16 digit second-order precision.

Ranking	*Method*	*Succ*	*Fcnt*	*Accy*	*Time*
(BFGS) 1	CONJUGATE	25	154.5	-13.3	0.2
2	BUPDATE	25	156.4	-13.2	0.4
3	CHOLESKY	25	157.2	-13.2	0.3
4	HUPDATE	25	159.1	-13.1	0.3
(DFP) 1	CONJUGATE	21	20244.8	-12.4	30.0
2	HUPDATE	19	14476.7	-11.9	21.0
3	BUPDATE	19	17151.5	-12.3	42.0
4	CHOLESKY	19	22718.1	-12.2	52.9

Table 6.4: Standard line search and 16 digit second-order precision.

compared to 399 ± 11 for the DFP implementations. The mean execution times ranged from 0.4 to 0.6 seconds per test problem. Overall, all of the BFGS implementations produced very similar results, as did all of the DFP implementations.

Standard line search. The mean number of function evaluations ranged from 154.5 for the BFGS implementation of CONJUGATE through to 22718.1 for the DFP implementation of CHOLESKY. The mean number of function evaluations required by the BFGS implementations was 157 ± 3 compared to a massive 18600 ± 4200 for the DFP implementations. The number of function evaluations required by the DFP implementations increased dramatically with the standard line search—as expected due to the known instability of DFP methods with less accurate line searches. The big difference in the mean number of function evaluations was also reflected in the mean execution times which ranged from 0.2 to 52.9 seconds per test problem. The BFGS implementations with the standard line search used approximately half as many function evaluations as the BFGS implementations with the strict line search. This is a major reason for the

Ranking	*Method*	*Succ*	*Fcnt*	*Accy*	*Time*
(BFGS) 1	CONJUGATE	333	318.0	-13.8	0.4
2	CHOLESKY	328	360.6	-13.8	0.5
3	BUPDATE	292	332.2	-13.7	0.6
4	HUPDATE	268	312.7	-13.9	0.4
(DFP) 1	CONJUGATE	322	440.1	-14.0	0.5
2	CHOLESKY	315	467.1	-13.9	0.6
3	BUPDATE	287	416.7	-13.6	0.6
4	HUPDATE	248	396.8	-13.9	0.5

Table 6.5: Strict line search and 16–2 digit second-order precision.

Ranking	*Method*	*Succ*	*Fcnt*	*Accy*	*Time*
(BFGS) 1	CONJUGATE	332	158.3	-13.1	0.3
2	CHOLESKY	323	170.4	-13.0	0.4
3	BUPDATE	289	153.4	-12.8	0.4
4	HUPDATE	263	151.2	-13.0	0.3
(DFP) 1	CONJUGATE	206	16387.6	-12.0	31.8
2	CHOLESKY	205	18225.7	-12.2	44.0
3	BUPDATE	139	16650.8	-11.7	46.2
4	HUPDATE	138	17759.0	-12.1	34.3

Table 6.6: Standard line search and 16–2 digit second-order precision.

popularity of less accurate line searches, and of course why they have become standard.

Limited precision second-order information

The performance of each implementation as the precision of the second-order information varied from 16 to two digits with the strict and standard line searches is presented in Tables 6.5 and 6.6. Once again, when successful all implementations produced similarly accurate approximations to the solutions of the test problems.

Strict line search. The number of successfully solved test problems ranged from 333 for the BFGS implementation of CONJUGATE down to 248 for the DFP implementation of HUPDATE. The mean number of function evaluations ranged from 312.7 for the BFGS implementation of HUPDATE through to 467.1 for the DFP implementation of CHOLESKY. The mean number of function evaluations required by the BFGS implementations was 337 ± 24 compared to 432 ± 35 for the DFP implementations. The mean execution times ranged from 0.4 to 0.6 seconds per test problem. Overall, all of

the BFGS implementations produced very similar results, as did all of the DFP implementations. The results in Table 6.5 are presented graphically in Figures 6.1 and 6.2.

Standard line search. The number of successfully solved test problems ranged from 332 for the BFGS implementation of CONJUGATE down to 138 for the DFP implementation of HUPDATE. The mean number of function evaluations ranged from 151.2 for the BFGS implementation of HUPDATE through to 18225.7 for the DFP implementation of CHOLESKY. The mean number of function evaluations required by the BFGS implementations was 161 ± 10 compared to 17300 ± 1000 for the DFP implementations. The mean execution times ranged from 0.3 to 46.2 seconds per test problem. Once again the BFGS implementations with the standard line search used approximately half as many function evaluations as the BFGS implementations with the strict line search. The results in Table 6.6 are presented graphically in Figures 6.3 and 6.4.

Comparing Figures 6.1 and 6.2 it is clear that the BFGS implementations perform quite similarly to the DFP implementations when using the strict line search. This is not surprising given the 1972 result by Dixon [27, 28] which shows that BFGS implementations produce iterates which are *identical* to those produced by DFP implementations in exact arithmetic when used with an exact line search. Although the arithmetic is not exact and neither is the strict line search the accuracy is sufficient for the observed differences between the BFGS and DFP implementations with the strict line search to be minor. The same cannot be said for the standard line search. As shown in Figures 6.3 and 6.4 the BFGS implementations produced noticeably different results to the DFP implementations with the standard line search. The BFGS implementations maintained a stable performance with either line search. The performance of the DFP implementations deteriorated markedly with the standard line search. As shown in Figures 6.1–6.3 individual differences in the BFGS implementations (with either line search) and the DFP implementations (with the strict line search) do not become noticeable until the precision of the second-order information falls below about eight digits. However as shown in Figure 6.4 the DFP factorisation implementations produced noticeably better results at single precision (8 digits) than the non-factored DFP implementations.

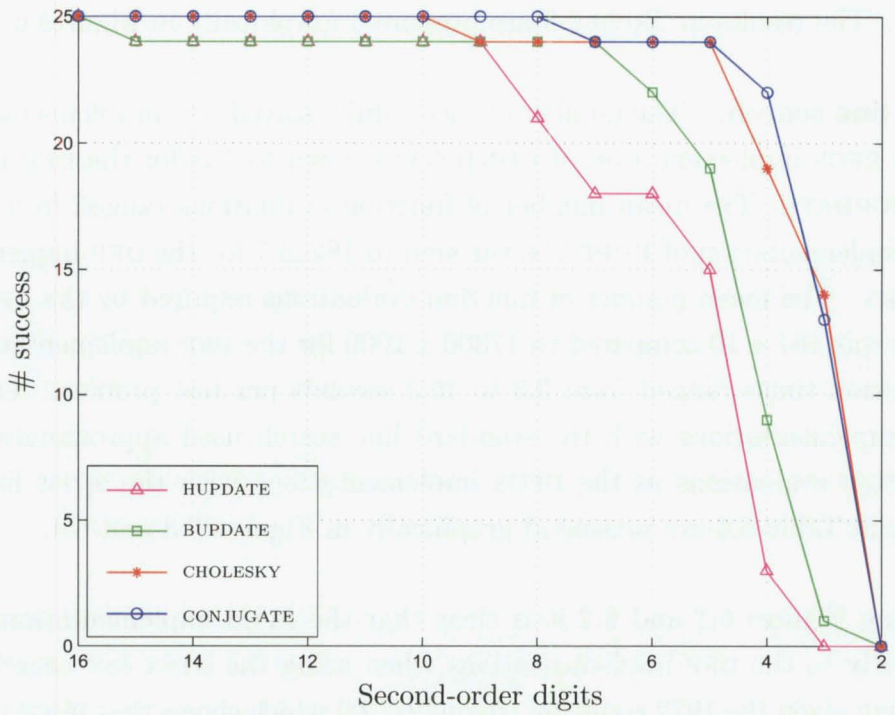

Figure 6.1: BFGS implementations with the strict line search.

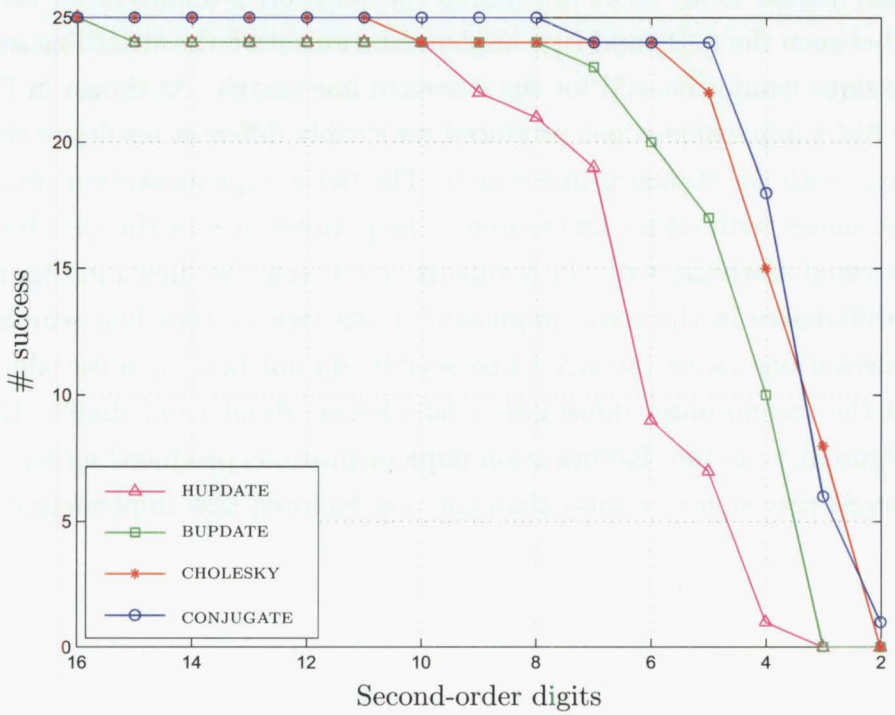

Figure 6.2: DFP implementations with the strict line search.

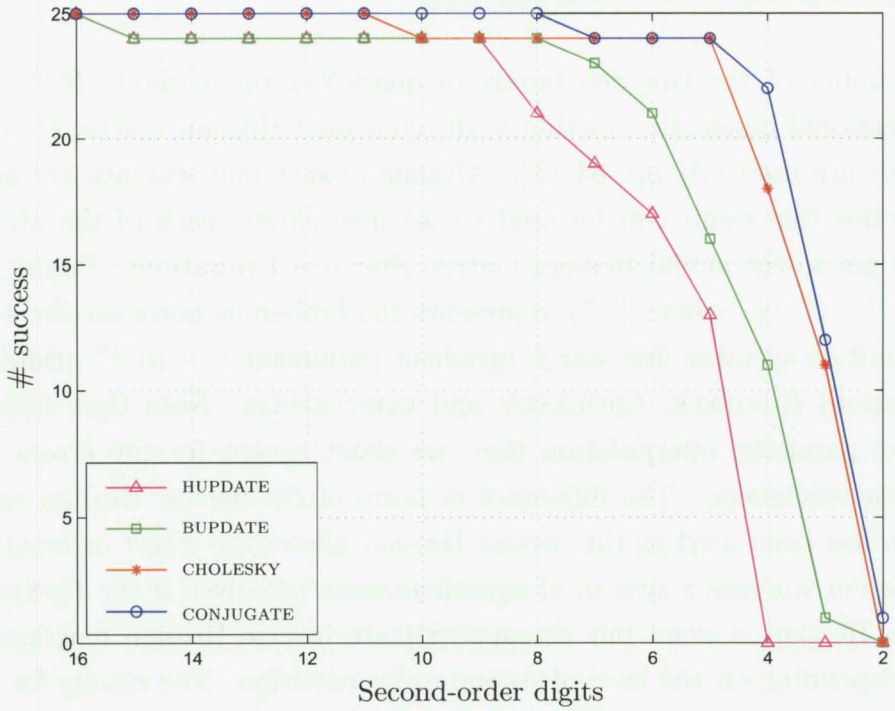

Figure 6.3: BFGS implementations with the standard line search.

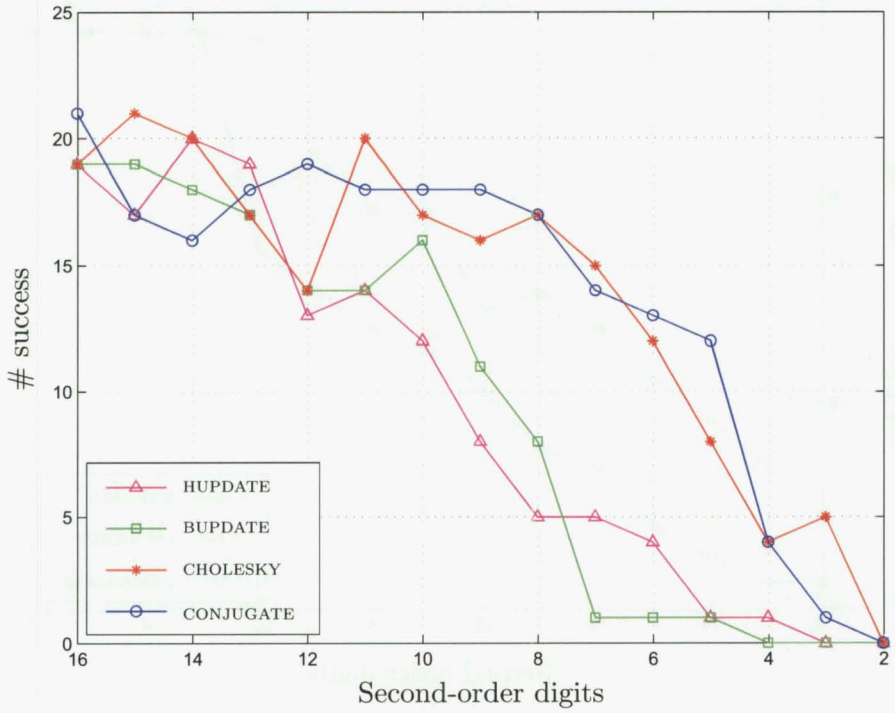

Figure 6.4: DFP implementations with the standard line search.

6.3 Quadratic termination

For any member of the Broyden family of quasi-Newton methods $B^{(n+1)} = G$ for any n-dimensional quadratic function with (constant) Hessian matrix G when exact line searches are used [31, pp. 64–65]. Although exact line searches are not carried out in practice this result can be used to see how closely each of the above implementations get to the actual Hessian matrix after $n + 1$ iterations. Figure 6.5 shows $\log_{10} \|H^{(n+1)} - G^{-1}\|_F$ where $\|\cdot\|_F$ represents the Frobenius norm for the 4-d Hilbert quadratic and an accurate line search (gradient parameter $\sigma = 10^{-10}$) using the BFGS implementations BUPDATE, CHOLESKY and CONJUGATE. Note that since the line searches use parabolic interpolation they are exact except for the errors due to finite precision arithmetic. The difference in norm of the inverse Hessian rather than the Hessian has been used as the inverse Hessian allows the direct calculation of the search direction whereas a system of equations must be solved if the Hessian is used. The inverse Hessian is exact but the approximate inverse Hessian matrices $H^{(k)}$ are truncated depending on the level of second-order precision. The results for HUPDATE clutter the figure somewhat and have been omitted. However if included the plot for

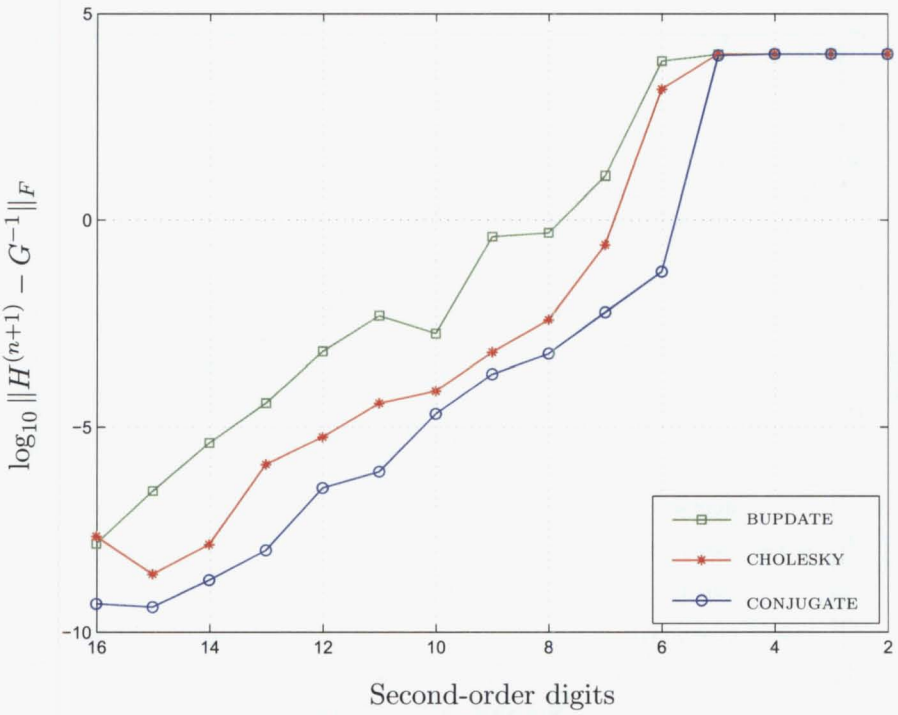

Figure 6.5: Difference in norm after $n+1$ iterations with varying second-order precision for the 4-d Hilbert quadratic.

HUPDATE would oscillate between the lines for CHOLESKY and CONJUGATE. The DFP implementations are not shown as they produce almost identical iterates with accurate line searches.

Note that as the precision of the second-order information falls below about five digits there is a plateau in Figure 6.5 with a height of about four. The height of this plateau coincides with the norm of the inverse Hessian of the 4-d Hilbert quadratic ($\log_{10} \|G^{-1}\|_F \approx 4.0146$). Presumably once the precision of the second-order information falls below a certain level there is insufficient information to approximate the inverse Hessian to any significant level. Similar results are produced with Hilbert quadratics of different dimensions. In higher dimensions the height of the plateau matches the norm of the inverse Hessian but the plateau starts at higher levels of second-order precision. In lower dimensions the plateau effect is lost and the differences in the performances of the implementations are reduced.

Chapter 7

Summary and concluding remarks

A brief introduction along with background theory on optimisation and positive bases was presented in Chapters 1 and 2. Two algorithm templates for a class of provably convergent frame-based line search algorithms for unconstrained optimisation that do not explicitly rely on derivative information was then presented in Chapter 3. Although these algorithm frameworks were established in [65] an alternate convergence proof was presented for the simple descent algorithm and new convergence results establishing the conditions necessary for convergence of the full sequence of iterates generated by the opportunistic sufficient descent algorithm were developed. The new convergence results for the opportunistic algorithm show that derivative information is required to estimate a suitable lower bound on the sufficient descent parameter. However it was shown that this reliance on derivative information is in the weakest possible sense since any estimate which overestimates the size of the gradient whilst maintaining a $o(h^{(k)})$ sufficient descent parameter suffices. Furthermore this lower bound can be determined without the need to calculate any additional information. An updated bound is only required when the current sufficient descent parameter is too large. A frame must be completed about the current iterate whenever this occurs and so enough information is available to revise the sufficient descent parameter. It was also shown that with a frame-based approach the error in the estimate of the lower bound tends to zero as the frame size tends to zero so that a self-correcting sequence of sufficient descent parameters could be determined automatically by the algorithm.

Using the theory for aligning positive bases with nearby constraints established in [66] both of the frame-based line search templates for unconstrained optimisation were extended for linearly constrained optimisation in Chapter 4 and full convergence results were presented. To maintain focus on the algorithm frameworks only a single, simple, forward-tracking line search was implemented with each of the algorithm templates. However it is important to emphasize that many other types of line search are possible.

A potential weakness with a common choice of stopping conditions for pattern search, grid- and frame-based methods was identified in Chapter 5. It was shown that a poor choice of grid can produce GLMs an arbitrary distance from the minimiser of even a strictly convex quadratic function in just two dimensions. This has serious implications for practical grid- and frame-based algorithms that terminate at the equivalent of a GLM whenever the grid or frame size parameter is reduced below some cut-off value.

Several examples were presented showing it is possible for pattern search, grid- and frame-based algorithms to return arbitrarily bad approximations to stationary points of the objective function. Furthermore an algorithm may rapidly (and prematurely) reduce the mesh, grid, or frame size parameter if a sequence of GLMs is located far from the minimiser. Without an appropriate compensatory strategy the convergence of such algorithms, although guaranteed in theory, may be extremely poor in practice. It was shown that such problems do not occur with grids based on conjugate directions. Conjugate grids guarantee that *any* GLM will be within one cell diameter of the minimiser of a strictly convex quadratic function. Under such circumstances it is appropriate to use the grid size parameter to determine when an algorithm may be terminated. Similar benefits are expected for non-quadratic functions with grid directions aligned with conjugate directions of an appropriate approximating quadratic function (see [20] for a discussion on generating sets of conjugate directions). Although there is a lower bound on the smallest angle between pairs of conjugate directions for strictly convex quadratic functions care must be taken when the Hessian matrix of an approximating quadratic function is singular, or nearly so, at or near a minimiser. In this situation it is possible for a pair of conjugate directions to become arbitrarily close to linearly dependent. Some limited numerical results showing the improved performance of a grid-based method using conjugate grid directions are presented in [20].

The use of conjugate directions when derivative information is available was investigated in Chapter 6. The performance of four BFGS and four DFP quasi-Newton implementations (including a conjugate directions factorisation) was examined using a suite of 25 test functions and two line searches (strict and standard) as the precision of second-order information was varied from 16 to two digits. It was shown that when second-order information was available to double precision (16 digits) there was no observable advantage in any particular implementation. If second-order information was only available to single precision (8 digits) then a factorisation strategy greatly improved the performance of the DFP implementations with the weak line search but did not greatly alter the performance of the BFGS implementations (with either line search). Although the BFGS implementations with the strict line search were slightly more robust than the BFGS implementations with the standard line search they required nearly double the number of function evaluations. When second-order information was available to at least single precision (8 digits) there was no real advantage in any particular BFGS implementation. However it was shown that conjugate factorisation implementations possessed good numerical stability properties when approximate second-order information was available to only limited precision.

Cholesky factorisation implementations enable second-order information to be updated and a new search direction computed in $\mathcal{O}(n^2)$ operations per iteration as well as allowing the easy detection of loss of positive definiteness of the second-order matrices. However the use of conjugate factorisations eliminates the possibility of negative definiteness or indefiniteness of the inverse approximate Hessian matrices whilst maintaining $\mathcal{O}(n^2)$ operations efficiency at each iteration. Furthermore the conjugate factorisation implementation produced better approximations to the inverse Hessian matrices of n-dimensional Hilbert quadratics when terminated after $n+1$ iterations than the other methods. As the precision of the second-order information was reduced the conjugate factorisation implementation was able to maintain accurate approximations to the inverse Hessian longer than the other methods. The conjugate factorisation implementation successfully solved more test problems in significantly fewer function evaluations than any of the other implementations, including Cholesky factorisation. Conjugate direction factorisations should be of practical importance when gradient information is approximated as it is likely that second-order information would only be available (reliable) to limited precision.

7.1 Where to next?

Some areas that could benefit from further research are now highlighted.

Numerical validation

As noted in Chapter 1 extensive performance results for specific implementations of algorithms conforming to the frameworks developed in this thesis are not presented. While this is an important consideration such an investigation would have shifted the focus of the research away from the theoretical frameworks and onto specific implementations; but practical algorithm development needs to be accompanied by performance-based testing. Simply ensuring a parameter is positive may be sufficient to establish a particular theoretical result but determining a suitable value in practice may depend on extensive numerical testing [8].

Line searches

The frame-based line search algorithms presented in this thesis all used the same simple line search. As shown in [65] many other line search options are possible. Furthermore, requiring the monotonic decrease of the function values severely restricts the points

which can be considered. A (safe-guarded) non-monotone line search strategy may enable an algorithm to step over regions that are difficult for monotone line searches to successfully navigate around. Clearly, incorporating and comparing such different line search strategies would produce a large amount of test data, the examination of which is beyond the scope of this thesis. Non-monotone line search strategies and the so-called *watchdog* technique are discussed in [12, 34, 42].

Aligned positive bases

The only requirements on the members of the aligned positive bases u_q is that each u_q points into the feasible region of constraint c_{i_q} and is orthogonal to all other constraint normals in the current active constraint working set. However to enforce the maximum length condition required for convergence it may be necessary to retrospectively scale these vectors. If, for particularly bad choices of u_q, the scaled product $u_q^{\mathsf{T}} a_{i_q}$ is evaluated as zero in finite precision arithmetic then the scaled u_q would *appear orthogonal* to the constraint normal a_{i_q}. There may be better ways of determining the aligned positive bases that avoid such problems. Such considerations, although important, are beyond the scope of this thesis.

Conjugate grids

The danger of relying solely on the standard stopping criteria for pattern search, grid- and frame-based algorithms was established in Chapter 5. It was shown that GLMs are guaranteed to be within one cell diameter of the minimiser of a strictly convex quadratic function whenever the grid directions are aligned with the conjugate directions of the quadratic function. The ability to efficiently generate such conjugate grids is therefore very important. However further investigation of this topic is beyond the scope of this thesis.

Hybrid methods

The material presented in this thesis leads to the possibility of creating hybrid methods that are able to usefully exploit derivative information if it becomes available (and reliable) but that also work well in practice when derivative information is either unavailable or unreliable. However further investigation of this topic is beyond the scope of this thesis.

Clearly much work remains to be done ...

Acknowledgements

I would like to thank my research supervisors Dr Chris Price and Associate Professor Ian Coope for their guidance during this project.

This research was financially supported by a Top Achiever Doctoral Scholarship.

References

[1] E. J. Anderson and M. C. Ferris. A direct search algorithm for optimization with noisy function evaluations. *SIAM Journal on Optimization*, 11(3):837–857, 2001.

[2] E. M. L. Beale. *Introduction to optimization.* John Wiley & Sons, Chichester, 1988.

[3] M. J. Box, D. Davies, and W. H. Swann. Non-linear optimization techniques. Monograph 5, Imperial Chemical Industries Limited, Edinburgh, 1969.

[4] K. W. Brodlie, A. R. Gourlay, and J. Greenstadt. Rank-one and rank-two corrections to positive definite matrices expressed in product form. *Institute of Mathematics and its Applications Journal*, 11(1):73–82, February 1973.

[5] C. G. Broyden. Quasi-Newton methods and their application to function minimisation. *Mathematics of Computation*, 21(99):368–381, July 1967.

[6] C. G. Broyden. The convergence of a class of double-rank minimization algorithms. Part 1: General considerations. *Institute of Mathematics and its Applications Journal*, 6(1):76–90, March 1970.

[7] C. G. Broyden. The convergence of a class of double-rank minimization algorithms. Part 2: The new algorithm. *Institute of Mathematics and its Applications Journal*, 6(3):222–231, September 1970.

[8] D. Byatt. Convergent variants of the Nelder-Mead algorithm. Master's thesis, University of Canterbury, Christchurch, New Zealand, 2000.

[9] D. Byatt, I. D. Coope, and C. J. Price. Performance of various BFGS and DFP implementations with limited precision second order information. Research report UCDMS2003/1, University of Canterbury, Christchurch, New Zealand, January 2003.

[10] D. Byatt, I. D. Coope, and C. J. Price. Conjugate grids for unconstrained optimisation. *Computational Optimization and Applications*, 29(1):49–68, October 2004.

[11] D. Byatt, I. D. Coope, and C. J. Price. Performance of various BFGS implementations with limited precision second-order information. *ANZIAM Journal*, 45(4):511–522, April 2004.

[12] R. M. Chamberlain, M. J. D. Powell, C. Lemarechal, and H. C. Pedersen. The watchdog technique for forcing convergence in algorithms for constrained optimization. *Mathematical Programming Study*, 16:1–17, 1982.

[13] T. D. Choi and C. T. Kelley. Superlinear convergence and implicit filtering. *SIAM Journal on Optimization*, 10(4):1149–1162, 2000.

[14] A. R. Conn, N. I. Gould, and Ph. L. Toint. *Trust-region methods.* MPS/SIAM series on optimization. Society for Industrial and Applied Mathematics and Mathematical Programming Society, Philadelphia, 2000.

[15] A. R. Conn, K. Scheinburg, and Ph. L. Toint. On the convergence of derivative-free methods for unconstrained optimization. In M. D. Buhmann and A. Iserles, editors, *Approximation theory and optimization*, pages 83–108. Cambridge University Press, Cambridge, 1997.

[16] A. R. Conn, K. Scheinburg, and Ph. L. Toint. Recent progress in unconstrained nonlinear optimization without derivatives. *Mathematical Programming*, 79:397–414, 1997.

[17] I. D. Coope. A conjugate direction implementation of the BFGS algorithm with automatic scaling. *Journal of the Australian Mathematics Society Series B*, 31:122–134, 1989.

[18] I. D. Coope. Personal communication, September 2003. Effects of noise on the calibration of robotic arms.

[19] I. D. Coope. Personal communication, January 2004. Example of a sequence of degenerate positive bases.

[20] I. D. Coope and C. J. Price. A direct search conjugate directions algorithm for unconstrained minimization. *ANZIAM Journal*, 42(E):C478–C498, 2000. `http://anziamj.austms.org.au/V42/CTAC99/`.

[21] I. D. Coope and C. J. Price. Frame based methods for unconstrained optimization. *Journal of Optimization Theory and Applications*, 107(2):261–274, November 2000.

[22] I. D. Coope and C. J. Price. On the convergence of grid-based methods for unconstrained optimization. *SIAM Journal on Optimization*, 11(4):859–869, 2001.

[23] I. D. Coope and C. J. Price. Positive bases in numerical optimization. *Computational Optimization and Applications*, 21:169–175, 2002.

[24] W. C. Davidon. Optimally conditioned optimization algorithms without line searches. *Mathematical Programming*, 9(1):1–30, August 1975.

[25] W. C. Davidon. Variable matric method for minimization. *SIAM Journal on Optimization*, 1(1):1–17, February 1991. Originally published without the preface as Argonne National Laboratory Research and Development Report 5990, May 1959.

[26] C. Davis. Theory of positive linear dependence. *American Journal of Mathematics*, pages 733–746, 1954.

[27] L. C. W. Dixon. Quasi-Newton algorithms generate identical points. *Mathematical Programming*, 2:383–387, 1972.

[28] L. C. W. Dixon. Quasi-Newton techniques generate identical points II: The proofs of four new theorems. *Mathematical Programming*, 3(3):345–358, 1972.

[29] E. D. Dolan, R. M. Lewis, and V. J. Torczon. On the convergence of pattern search. *SIAM Journal on Optimization*, 14(2):567–583, 2003.

[30] R. Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317–322, August 1970.

[31] R. Fletcher. *Practical methods of optimization.* John Wiley & Sons, New York, second edition, 1987.

[32] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *The Computer Journal*, 6(2):163–168, July 1963.

[33] L. R. Foulds. *Optimization techniques.* Springer-Verlag, New York, 1981.

[34] M. G. Gasparo, A. Papini, and A. Pasquali. Nonmonotone algorithms for pattern search methods, September 2000.

[35] P. E. Gill and W. Murray. Quasi-Newton methods for unconstrained optimization. *Institute of Mathematics and its Applications Journal*, 9(1):91–108, 1972.

[36] P. E. Gill and W. Murray. Modification of matrix factorizations after a rank one update. In D. Jacobs, editor, *The state of the art in numerical analysis*, pages 55–83, London, 1976. Institute of Mathematics and its Applications, Academic Press.

[37] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization.* Academic Press, London, 1981.

[38] D. Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24(109):23–26, January 1970.

[39] L. Grandinetti. Factorization versus non-factorization in quasi-Newtonian algorithms for differentiable optimization. In *Third Symposium on Operations Research (University of Mannheim, Mannheim, 1978)*, pages 255–274, Konigstein, 1979. Hain. Section I.

[40] L. Grandinetti. Factorized variable metric algorithms for unconstrained optimization. In K. Iracki, K. Malanowski, and S. Walukiewicz, editors, *Optimization Techniques. Proceedings of the 9th IFIP Conference on Optimization Techniques, Warsaw, September 4–8, 1979*, volume 23 of *Lecture Notes in Control and Information Sciences*, pages 52–61, Berlin, September 1979. Springer-Verlag.

[41] J. Greenstadt. Variations on variable-metric methods. *Mathematics of Computation*, 24(109):1–22, January 1970.

[42] L. Grippo, F. Lampariello, and S. Lucidi. A nonmonotone line search technique for Newton's method. *SIAM Journal on Numerical Analysis*, 23(4):707–716, August 1986.

[43] Harwell subroutine library, release 10, Advanced Computing Department, AEA Industrial Technology, Harwell Laboratory, Oxfordshire, UK, 1990.

[44] R. Hooke and T. A. Jeeves. Direct search solution of numerical and statistical problems. *Journal of the Association for Computing Machinery (ACM)*, 8(2):212–219, April 1961.

[45] T. G. Kolda, R. M. Lewis, and V. J. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482, 2003.

[46] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1):112–147, 1998.

[47] R. M. Lewis and V. J. Torczon. Rank ordering and positive bases in pattern search algorithms. Technical Report 98-31, Institute for Computer Applications in Science and Engineering, NASA Langley Research Centre, Hampton, VA, 1998.

[48] R. M. Lewis and V. J. Torczon. Pattern search algorithms for bound constrained optimization. *SIAM Journal on Optimization*, 9(4):1082–1099, 1999.

[49] R. M. Lewis and V. J. Torczon. Pattern search methods for linearly constrained minimization. *SIAM Journal on Optimization*, 10(3):917–941, 2000.

[50] R. M. Lewis, V. J. Torczon, and M. W. Trosset. Direct search methods: then and now. *Journal of Computational and Applied Mathematics*, 124(1–2):191–207, 2000.

[51] S. Lucidi and M. Sciandrone. On the global convergence of derivative-free methods for unconstrained optimization. *SIAM Journal on Optimization*, 13(1):97–116, 2002.

[52] MATLAB. The MathWorks Inc, 24 Prime Park Way, Natick, Massachusetts, USA. `http://www.mathworks.com`.

[53] K. I. M. McKinnon. Convergence of the Nelder-Mead simplex method to a non-stationary point. *SIAM Journal on Optimization*, 9(1):148–158, 1998.

[54] J. J. Moré. Notes on optimization software. In M. J. D. Powell, editor, *Non-linear optimization 1981*, pages 339–352. Academic Press, New York, 1982.

[55] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, 7(1):17–41, March 1981.

[56] D. D. Morrison. Optimization by least squares. *SIAM Journal on Numerical Analysis*, 5:83–88, 1968.

[57] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.

[58] J. Nocedal and S. J. Wright. *Numerical optimization.* Springer, New York, 1999.

[59] M. R. Osborne and M. A. Saunders. Descent methods for minimization. In R. S. Anderssen, L. D. Jennings, and D. M. Ryan, editors, *Optimization*, pages 221–237. University of Queensland Press, St. Lucia, 1972.

[60] M. J. D. Powell. Updating conjugate directions by the BFGS formula. *Mathematical Programming*, 38(1):29–46, 1987.

[61] M. J. D. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, 7:287–336, 1998.

[62] C. J. Price. Personal communication, July 2002. Comment on the number of grid local minimisers in higher dimensions.

[63] C. J. Price. Personal communication, April 2004. Comment on positive bases.

[64] C. J. Price. Personal communication, April 2004. Comment on structural equivalence and conjugate directions.

[65] C. J. Price and I. D. Coope. Frame based ray search algorithms in unconstrained optimisation. *Journal of Optimization Theory and Applications*, 116(2):359–377, February 2003.

[66] C. J. Price and I. D. Coope. Frames and grids in unconstrained and linearly constrained optimization: a non-smooth approach. *SIAM Journal on Optimization*, 14(2):415–438, 2003.

[67] C. J. Price, I. D. Coope, and D. Byatt. A convergent variant of the Nelder-Mead algorithm. *Journal of Optimization Theory and Applications*, 113(1):5–19, April 2002.

[68] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, October 1960.

[69] R. B. Schnabel and E. Eskow. A revised modified Cholesky factorization algorithm. *SIAM Journal on Optimization*, 9(4):1135–1148, 1999.

[70] H. A. Schwarz. Über ein die Flächen kleinsten Flächeninhalts betreffendes Problem der Variationsrechnung. *Acta Soc. Scient. Fen.*, 15:315–362, 1885. Reprinted in *Gesammelte Mathematische Abhandlungen*, 1:224–269, 1972.

[71] D. F. Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24(111):647–656, July 1970.

[72] A. H. Sherman. On Newton iterative methods for the solution of systems of nonlinear equations. *SIAM Journal on Numerical Analysis*, 15:755–771, 1978.

[73] V. J. Torczon. On the convergence of the multidirectional search algorithm. *SIAM Journal on Optimization*, 1(1):123–145, February 1991.

[74] V. J. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1):1–25, February 1997.

[75] J. van Tiel. *Convex analysis.* John Wiley & Sons, 1984.

[76] M. A. Woodbury. Inverting modified matrices. Statistical Research Group Report 42, Princeton University, Princeton, NJ, USA, 1950.

[77] M. H. Wright. Direct search methods: once scorned, now respectable. In D. F. Griffiths and G. A. Watson, editors, *Numerical Analysis 1995*, number 344 in Pitman Research Notes in Mathematics Series, pages 191–208, Harlow, UK, 1996. Addison Wesley Longman Ltd.

[78] Wen-Ci Yu. Positive basis and a class of direct search techniques. *Scientia Sinica (Zhongguo Kexue)*, pages 53–68, 1979. Special issue 1 on mathematics.

Index

U

V

W